НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК 004.891.2

«До захисту допущено»

Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
“___”_____2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: Система розумного міста на базі нейронних мереж _____
_____
_____

Виконав: студент другого курсу, групи ІТ-84мп_
(шифр групи)
_____Роспопа Павло Петрович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник ст. викладач, к.т.н. Сирота О.П. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант НК к.т.н, доцент Пасько В. П. _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент доцент, к.т.н., доцент каф. АСОІУ Ліщук К.І. _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Рівень вищої освіти – другий (магістерський)

Спеціальність 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ І.Р. Пархомей

(підпис)

«___»_____2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Роспопі Павлу Петровичу

(прізвище, ім'я, по батькові)

1. Тема дисертації «Система розумного міста на базі нейронних мереж»
науковий керівник дисертації ст.викладач, к.т.н. Сирота О.П.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 28 » жовтня 2019 р. № 3770-с
2. Термін подання студентом дисертації 18.11.19

3. Об'єкт дослідження – пошук оптимального маршруту в межах міста

4. Предмет дослідження – алгоритми машинного навчання для пошуку оптимального маршруту в місті

5. Перелік завдань, які потрібно розробити – аналіз предметної області та існуючих рішень; аналіз алгоритмів машинного навчання для знаходження оптимального маршруту; розробка нейронної мережі; дослідження ефективності навчання нейронної мережі знаходженню оптимального маршруту.

6. Орієнтовний перелік ілюстративного матеріалу – шість плакатів

7. Орієнтовний перелік публікацій – одна публікація

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Перевірка на співпадіння	Лісовиченко О.І., к.т.н., доцент кафедри ТК		
Нормконтроль	Пасько В.П., к.т.н., доцент кафедри ТК		

9. Дата видачі завдання 25.10.18

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області	28 жовтня 2018 р.	
2	Постановка задачі	12 грудня 2018 р.	
3	Аналіз існуючих рішень	25 січня 2019 р.	
4	Аналіз алгоритмів машинного навчання для пошуку оптимального маршруту	25 квітня 2019 р.	
5	Розробка нейронної мережі	20 травня 2019 р.	
6	Кероване навчання нейронної мережі за допомогою набору тестових даних	19 вересня 2019 р.	
7	Маркетинговий аналіз стартап-проекту	10 листопада 2019 р.	
8	Висновки	15 листопада 2019 р.	

Студент _____ П.П. Роспопа
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ О.П. Сирота
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

У роботі розглянуто проблему пошуку оптимального маршруту в межах міста.

Розроблено систему розумного міста, яка взаємодіє з об’єктами міста та застосовує алгоритми машинного навчання для автоматизації процесів, прогнозування подій та вирішення задач міста. На базі системи реалізований модуль Керування автомобільним трафіком, який використовує дані датчиків, камер відеоспостереження та інформацію про місцезнаходження користувачів для відображення актуального стану завантаженості доріг та дозволяє побудувати оптимальний маршрут в місті.

Розділ аналізу предметної області і постановки задачі розкриває аспекти концепції розумного міста. У розділі досліджені існуючі рішення, проведене комплексне порівняння наявних систем розумного міста. Сформульовані вимоги до системи.

Розділ аналізу використаних інформаційних технологій описує особливості технологій, які були застосовані при розробці системи.

У розділі розробки системи розкриті питання проектування та імплементації. Описані інструменти та методи розробки, особливості тестування.

Розділ маркетингового аналізу стартап-проекту містить комплексний аналіз поточної ситуації на ринку, стратегії та маркетингові плани впровадження системи.

Ключові слова: розумне місто, машинне навчання, нейронна мережа, аналітика великих даних, сервісна платформа.

Розмір пояснювальної записки – 80 аркушів, містить 12 ілюстрацій, 29 таблиць, 2 додатки.

ABSTRACT

The project examines the problem of finding optimal route in a city.

A smart city system that interacts with city objects and uses machine learning algorithms to automate processes, predict events and solve city problems has been developed. Based on capabilities of the system Traffic Management module was developed. It uses data of sensors, CCTV cameras and location information of users to model current state of traffic congestion and allows users to find optimal route in a city.

The chapter of subject area analysis and task formulation reveals aspects of the concept of a smart city. It explores existing solutions, comprehensively compares existent smart city systems and defines requirements for the system.

The chapter of analysis of used information technologies describes features of technologies that were used in the system.

The system development chapter addresses various design and implementation tasks. It contains information about used development best practices and tools. Describes testing phase of the system.

The marketing analysis chapter contains comprehensive analysis of the current market situation, describes strategies and marketing plans to bring the system to the market.

Keywords: Smart City, machine learning, neural network, big data analytics, service platform.

Explanatory note size – 80 pages, contains 12 illustrations, 29 tables, 2 applications.

Пояснювальна записка

до магістерської дисертації

на тему: *Система розумного міста на базі нейронних мереж*

Київ – 2019 року

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

HTTP – HyperText Transfer Protocol

API – Application Programming Interface

REST – Representational State Transfer

IoT – Internet of Things

DNN – Deep Neural Network

RL – Reinforcement Learning

ML – Machine Learning

HMM – Hidden Markov Model

ReLU – Rectified Linear Unit

GPU – Graphics Processing Unit

LSTM – Long Short-Term Memory

VAE – Variational Autoencoder

FDI – False Data Injection

BLE – Bluetooth Low Energy

CDN – Content Delivery Network

CLI – Command Line Interface

RPC – Remote Procedure Call

DSL – Domain-Specific Language

ВСТУП

У сучасному світі міста є ключовими економічними, адміністративними та географічними центрами суспільства. Найбільші мегаполіси за кількістю населення можуть перевищувати у декілька разів цілі країни. Так, згідно з прогнозами ООН, до 2050 року 68% населення планети буде проживати у містах. Стрімкі темпи урбанізації та розвиток промисловості змінюють підхід до планування і розвитку інфраструктури міст. Разом з цим зростають потреби населення. Це призводить до необхідності пошуку нових методів управління, нових засад взаємодії міста та його мешканців, переходу міста від формату «знеособленої території для виживання та задоволення базових потреб жителів» до формату «живої істоти», для якої люди є суб'єктами міського життя, з якими здійснюється інтерактивне спілкування.

Сучасні міста стикаються з численними проблемами, зокрема: погіршенням екологічної ситуації, складнощами в управлінні автомобільним трафіком на дорогах, великою кількістю машин у місті, неоптимальною організацією потоку людей. Намагаються розв'язати завдання боротьби зі злочинністю, ефективним використанням електроенергії та води, обробкою сміття. Для вирішення цих та багатьох інших задач планується використовувати передові інформаційні технології: технології Інтернету Речей, алгоритми машинного навчання та прогнозування, інфраструктуру хмарних обчислень.

У роботі розкрито суть системи розумного міста, яка завдяки використанню передових технологій об'єднує різні джерела даних для отримання комплексної картини стану процесів міста та дозволяє застосувати системний підхід для вирішення проблем. Описано особливості проектування та розробки модуля для пошуку оптимального маршруту в межах міста, що дозволить вирішити складнощі в логістиці перевезень вантажів та пересуванні мешканців міста. Проведений комплексний аналіз наявних рішень та розроблено якісно новий метод, що базується на використанні алгоритмів машинного навчання та нейронних мереж.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1 Концепція розумного міста

Міста – сучасні економічні центри. 55% ВВП та понад 60% економічного зростання формують метрополії з населенням понад 500 тис. За прогнозами ці показники будуть зростати в найближчі роки. Значний вплив на економіку, який створюють великі міста, показує дослідження, яке провела ОЕСР щодо відмінності в темпах зростання ВВП окремих територій залежно від відстані до міст [1]. На рис. 1.1 видно, що при віддаленні від міста показник ВВП території зменшується.

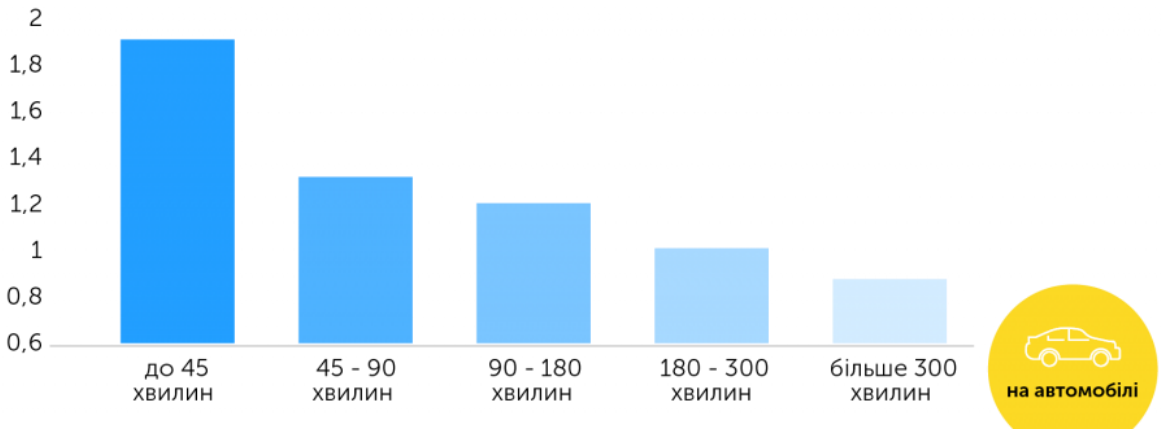


Рисунок 1.1. Різниця річного темпу приросту ВВП в залежності від відстані до великих міст, %

Крім економічної вигоди та більшої продуктивності, міста створюють платформу для стійкого і збалансованого зростання. Вони грають найважливішу роль у забезпеченні більш високого рівня добробуту їхніх жителів, у тому числі шляхом доступу до якісної освіти, можливостей працевлаштування, соціальної та громадянської активності.

Разом з перевагами, які надають міста, існують численні труднощі, які намагається вирішити влада міст. Більшість з них пов’язані зі стрімкими темпами урбанізації, розвитком промисловості, глобальною конкуренцією з боку міст за

залучення висококваліфікованих працівників та зростанням рівня очікувань резидентів міст [1]. Розглянемо їх в розрізі сфер діяльності:

1. Транспортна система. Характеризується втратами внаслідок перевантаженості доріг автомобілями, неефективним плануванням будівництва нової інфраструктури, великою кількістю ДТП.
2. Енергетика, вода та переробка відходів. Головними проблемами цих сфер господарства є неефективні системи розподілу, а також тривалий період відновлення після антропогенних чи природніх катастроф. Перехід на відновлювані джерела енергії відбувається дуже повільно. В більшості міст використовуються екологічно неефективні методи переробки відходів.
3. Безпека і захист даних. Спостерігається збільшення кількості кібератак на критично важливі об’єкти інфраструктури та окремих жителів. Ведеться боротьба зі злочинністю, соціальними хвилюваннями та тероризмом. Створюються механізми захисту конфіденційної інформації жителів міст.
4. Адміністративні послуги. Сфера відрізняється складністю у персоналізації послуг громадського користування для окремого жителя, дисбалансом попиту та пропозиції на послуги. Навігація по законодавчій базі є складною.
5. Громадський простір. Сфера характеризується браком даних для правильного міського планування, неефективним управлінням енергоспоживання будівель, неоптимальною організацією потоку людей та обмеженою кількістю стоянок у місцях загального користування.
6. Робочі місця та економічний розвиток. Чимало міст мають слаборозвинуту стартап екосистему. Це гальмує пошук та реалізацію інновацій та обмежує доступ до фінансування розумних проектів. Також варто зазначити обмежені можливості для пошуку нових вакансій.
7. Екологія. Внаслідок збільшення кількості жителів, автомобілів та підприємств екологічна ситуація погіршується. Влада міст займається питаннями розроблення норм шкідливих викидів, моніторингу рівня забруднення повітря та води.

В останні роки для вирішення вищезазначених проблем влада деяких міст почала застосовувати інформаційні технології. В першу чергу вони дозволили ефективно збирати інформацію, на основі якої можна приймати кращі рішення, будувати стратегію розвитку міста, займатись управлінням та плануванням. Проте, важливо зазначити, що технології самі по собі не можуть розв'язати всі соціальні, логістичні чи екологічні проблеми міст. Успішність спільної роботи приватних компаній та органів правління визначає наскільки ефективно міста можуть трансформуватися. Щоб міська влада та бізнес успішно співпрацювали в питанні розумних і стійких рішень містобудування, жителі міста також повинні бути в центрі цих процесів. Жодна з перерахованих зацікавлених сторін не існує у вакуумі. Всі стейкхолдери зацікавлені в досягненні єдиного результату: кращої якості життя, збільшення продуктивності і добробуту міста. У цьому випадку державно-приватне партнерство може стати основою розвитку міст. На рис. 1.2 показано структуру розумного міста, що охоплює послуги, які надає місто, інформаційні технології, завдяки яким можлива реалізація розумних рішень в різних сферах господарства, та стейкхолдерів, які зацікавлені в розвитку та трансформації міста.

Для досягнення головної мети – покращення умов життя населення міста – будь-яка технологічна ініціатива повинна дотримуватись принципів розумного міста, які є базисом у створенні й управлінні всією розумною інфраструктурою. До них відносять:

1. Інформаційна орієнтованість. Розумне рішення повинно ефективно здійснювати збір, інтеграцію, доповнення, зберігання та поширення даних для вдосконалення процесу прийняття рішень.
2. Взаємопов'язаність. Використання технологій Інтернету Речей для покращення можливостей міста сприймати та реагувати на нагальні потреби мешканців.
3. Інтелектуальність. Використання аналітики та когнітивних інструментів для прогнозування поведінкових змін.

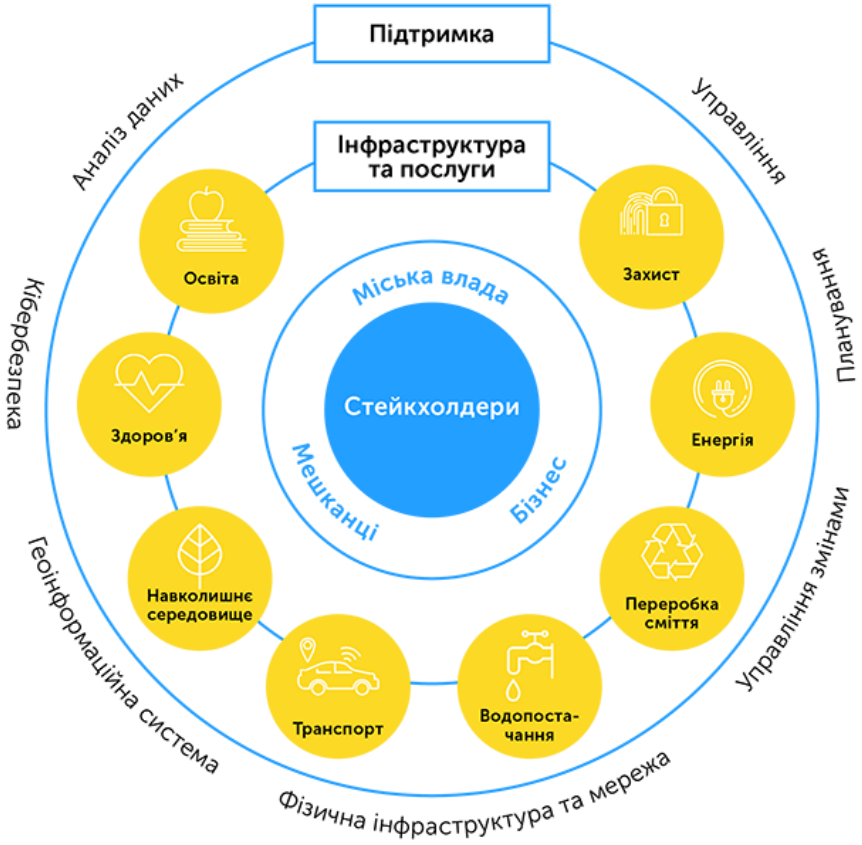


Рисунок 1.2. Структура розумного міста

4. Об'єднаність. Визначає потребу у співпраці мешканців та міських службовців, відкритості даних.
5. Зручність. Надання мешканцям міста доступних, зручних та ефективних послуг.

Зрілість розумного міста визначається станом розвитку базової інфраструктури та впровадженням технологічних інновацій. Структурно розумне місто можна поділити на 4 рівні [2]:

1. Рівень 1. Представляє собою базову інфраструктуру. Це фундаментальний чинник, без якого не можливий подальший розвиток міста. Включає в себе: транспорт, телекомунікації, енергетику, публічне освітлення, системи водопостачання та водовідведення.
2. Рівень 2. Відправна точка для розвитку міської інфраструктури Інтернету Речей. Складається з цифрових датчиків та розумних пристроїв, які

займаються збором та аналізом даних, зокрема поведінки користувачів та міжмашинної взаємодії.

3. Рівень 3. Формує платформу, яка аналізує та конкретизує дані, зібрані на нижчих рівнях. Являється фундаментом для підключення сторонніх додатків, які вирішують різноманітні проблеми міста.
4. Рівень 4. Програмні застосунки, які підключаються до платформи розумного міста та здійснюють контакт з кінцевими користувачами надаючи послуги.

Однією з поширених проблем міст є нерівномірне впровадження розумних рішень. Це часто пов’язано з повільним бюрократичним апаратом органів управління та призводить до фрагментації технологічного стеку та недоотримання максимальної користі від використовуваних технологій. Організації не можуть отримати повну цінність даних, якщо вони зберігаються в окремих системах і базах даних з обмеженим доступом та мають різний формат. Без можливості обмінюватися ключовою інформацією в режимі реального часу компанії, що працюють як в приватному, так і в державному секторі, не можуть розробляти програми, що автоматизують процеси та формують розумні можливості міста та його інфраструктури. Це також становить ризик для довгострокової безпеки життєдіяльності міста. Для прикладу, впровадження ефективної енергорозподільчої системи в окремо взятому районі міста разом із використанням старих енергомереж та центрального енергозабезпечення може призвести до дисбалансу і перевантаження системи в іншій частині міста. А використання розумних світлофорів та систем інтелектуального керування автомобільним трафіком, встановлених різними провайдерами та виходячи з різних технічних завдань, можуть призвести до створення «вузьких горлечок» – ділянок дороги з щільним трафіком, а отже зниження ефективності громадського транспорту та підвищення ризику виникнення аварійних ситуацій [3].

Для вирішення вищезазначених проблем міські органи управління повинні інтегрувати в різні області господарства міста єдину технологічну платформу,

якою можна буде централізовано керувати. Ця платформа повинна оперувати даними в одному стандарті, мати взаємопов’язані потоки та сховища даних та дозволяти стороннім розумним рішенням підключатись до неї за допомогою відкритих API (прикладних програмних інтерфейсів). При цьому основним завданням міста буде підтримка платформи та її масштабування під нові модулі архітектури розумного міста.

1.2 Об’єкт та предмет дослідження

Дана робота стосується розвитку області розумних міст: проектування та розробки інтелектуальних систем, що використовують алгоритми машинного навчання, нейронні мережі для автоматизації процесів, прогнозування подій та вирішення задач міста.

Прикладна частина роботи полягає в створенні модуля системи розумного міста для пошуку оптимального маршруту в межах міста та реалізації прикладного програмного інтерфейсу для надання доступу розробникам клієнтських застосунків до можливостей розробленої системи.

Об’єкт дослідження – методи пошуку оптимального маршруту в межах міста.

Предмет дослідження – алгоритми машинного навчання для знаходження оптимального маршруту в місті.

1.3 Огляд існуючих рішень

Поняття «розумне місто» охоплює різні сфери господарства міста. Завдяки використанню передових технологій комунікації та обробки інформації можна автоматизувати процеси та вирішити більшість проблем цих сфер. Розглянемо найбільш перспективні напрями розвитку розумного міста.

Розумне публічне освітлення (Smart Lighting). Рішення в даній області дозволяють економити електроенергію завдяки використанню світлодіодів,

зменшити витрати на ремонт та обслуговування, зменшити світлове забруднення. Стає можливим централізоване керування та розумне регулювання інтенсивності світла в залежності від погодних умов чи конкретних потреб місцевості. Крім ефективного виконання основної функції, даний тип систем може здійснювати моніторинг якості повітря, аналіз звуку, відеоспостереження, надання Wi-Fi мережі завдяки встановленню додаткових датчиків та інших розумних пристроїв. Дані, зібрані датчиками, можуть використовуватись іншими підсистемами розумного міста: наприклад, для управління дорожнім рухом та вільними автомобільними стоянками завдяки камерам відеоспостереження та обробці відеопотоку інтелектуальними системами. Фізичну інфраструктуру можна використовувати для встановлення станцій зарядки електричних велосипедів та електромобілів.

Для реалізації проекту розумного публічного освітлення необхідним є використання наступних технологій:

- світлодіодні лампи;
- оптоволоконна інфраструктура;
- енергетична інфраструктура;
- бездротові мережі;
- хмарні моніторингові системи;
- мобільні пристрої для персоналу обслуговування та ремонту;
- датчики та розумні об’єкти (датчики освітленості, руху, звуку; пристрої для визначення рівня забрудненості повітря, камери відеоспостереження з функцією розпізнавання предметів).

Розумний будинок (Smart House). Дане рішення передбачає використання системи високотехнологічних пристроїв в оселі для найбільш комфортного проживання людей. Завдяки впровадженню сучасних технологій системи розумних будинків дозволяють ефективно керувати енергоспоживанням, вентиляційною системою, регулювати температуру та освітленість приміщень. Знижується вплив на навколишнє середовище за рахунок раціонального споживання енергії, води та оптимізації сортування і переробки відходів. Безпека

мешканців гарантується завдяки використанню датчиків руху, присутності, вібрації, розбиття скла, відкриття вікна або дверей, відеоспостереження, електронних замків, модулів управління воротами та сирен.

Основні технології, які застосовуються при реалізації розумного будинку:

- бездротові мережі;
- доповнена та віртуальна реальність, чат-боти;
- аналітичні інструменти контролю за використанням енергії;
- IoT-шлюзи, хмарні системи;
- датчики та пристрої: датчики руху, вібрації, відкриття вікна чи дверей, розумні вимикачі світла, модулі управління шторами та ролетами, контролери для управління світлодіодними світильниками, термостати для підтримки постійної температури або її автоматичного регулювання, терморегулятори для управління потужністю батарей опалення, гігростати для підтримки постійної вологості або її регулювання.

Інтелектуальні системи управління дорожнім рухом. Головним завданням систем управління дорожнім рухом є управління трафіком в містах, полегшення навігації, моніторинг завантаженості доріг у режимі реального часу та прокладання оптимального маршруту. Рішення в цій сфері сприяють оптимізації пересування людей та товарів, покращують економіку, суспільну безпеку та навколишнє середовище, а також логістичну галузь в цілому. Інтеграція з міським громадським транспортом дозволяє жителям міста отримувати актуальну інформацію розкладу, відстежувати будь-які зміни у напрямках руху, підвищуючи комфорт поїздки. Існують системи, які дозволяють контролювати порушення правил дорожнього руху та сповіщати про всі інциденти в правоохоронні органі.

В результаті впровадження систем управління транспортом спостерігається значне зменшення об’єму шкідливих викидів в атмосферу, зменшується кількість заторів, витрат пального за рахунок оптимізації маршрутів; скорочується час в дорозі, що, як наслідок, зменшує витрати на транспортацію вантажів та населення.

Розумна автомобільна стоянка. Згідно останніх досліджень одна третя трафіку автомобілів в центрі міста обумовлена пошуком водіями стоянки. Для того, аби більш раціонально керувати автомобільними стоянками та збільшити дохід за рахунок динамічного визначення вартості паркування, деякі міста почали впроваджувати інтелектуальні системи, які виявляють і направляють водіїв на вільні автомобільні стоянки та допомагають відстежувати правопорушення. Системи розумних автомобільних стоянок засновані на використанні датчиків та камер відеоспостереження, які визначають зайнятість стоянки та розпізнають номер автомобілів. Жителі міст можуть перевірити актуальний стан автомобільних стоянок, забронювати та оплатити місце через мобільний додаток. Таким чином, дане рішення допомагає зменшити кількість заторів у центрі міста та зонах з інтенсивним рухом, знизити витрати палива та шкідливі викиди в атмосферу.

Система розподілення електроенергії (Smart Grid). Для більш ефективного управління енергоресурсами та підвищення стійкості деякі міста впроваджують новітні енергосистеми, які використовують розумні датчики, інтелектуальні системи керування та підтримують децентралізацію виробництва енергії та P2P торгівлю енергоресурсами. Smart Grid системи дозволяють балансувати навантаження, динамічно реагувати на піки споживання енергії, проводити збір та аналіз даних системи у реальному часі, швидко виявляти несправності та відновлювати свою роботу після надзвичайної ситуації.

Державні онлайн-послуги. IT технології також успішно застосовуються в державному секторі для покращення якості послуг держави. Це включає в себе різноманітні веб-портали, мобільні додатки з розкладом громадського транспорту, системи відкритих даних, які працюють на базі блокчейн-мереж для захисту даних від несанкціонованої модифікації тощо.

Більшість існуючих проектів розумного міста спрямовані на вирішення конкретної проблеми сфери господарства. Це так звані вертикальні системи, які застосовують M2M підхід та використовують чітко визначені джерела даних.

Одним з прикладів вертикальної системи розумного міста є система керування автомобільними стоянками компанії ZagrebParking в місті Загреб. За допомогою мобільного додатку жителі можуть відстежувати наявність стоянок та купувати квитки. Громадські стоянки оснащені камерами, що дозволяють розпізнавати автомобільні номери і перевіряти, чи була здійснена оплата.

Для автоматизації процесів житлових будинків існують численні комерційні та безкоштовні системи, які застосовують принципи Інтернету Речей. Однією з таких систем є HomeGenie – серверний застосунок з відкритим кодом для домашньої автоматизації, створений для управління, контролю та моніторингу розумних пристроїв, підключених до Інтернету. Він забезпечує безперебійну інтеграцію з багатьма пристроями, сервісами, а також підтримує багато протоколів та технологій, таких як IR / RF Control та UPnP / DLNA. HomeGenie можна встановити на Windows, Linux та macOS. На рис. 1.3 зображений дашборд системи з віджетами, які відображають основну інформацію розумного будинку.

Основні переваги системи:

- API, зручний для розробників;
- вбудований редактор;
- засоби збору статистики;
- редактор віджетів.

На ринку електронних карт, рішень для управління автомобільним трафіком та технологій автономного керування транспортними засобами, які становлять особливо важливу частину концепції розумного міста, однією з провідних компаній є нідерландська компанія TomTom. Крім платних продуктів, компанія надає безкоштовний сервіс для відслідковування стану заторів у містах.

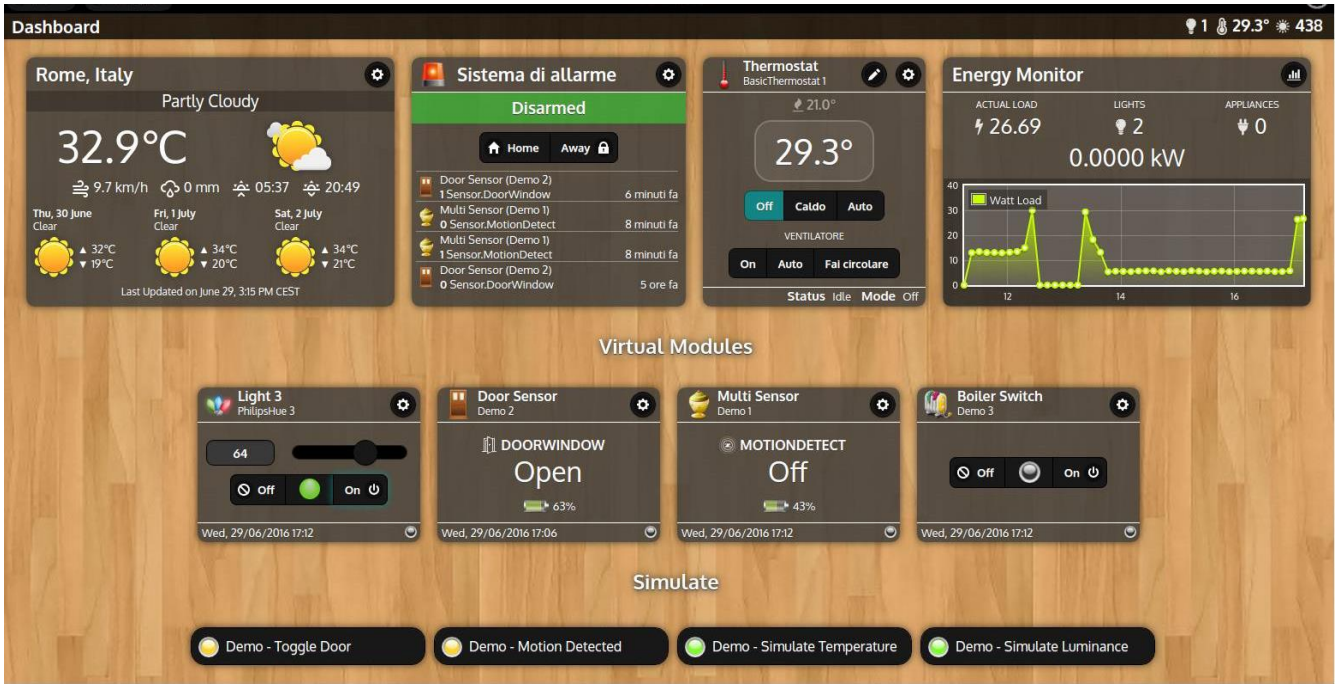


Рисунок 1.3. Дашборд HomeGenie

Платформа також містить інформацію про великі міста України, зокрема Київ. На рис. 1.4 показана головна сторінка сервісу з інформацією щодо транспортної ситуації в Києві.

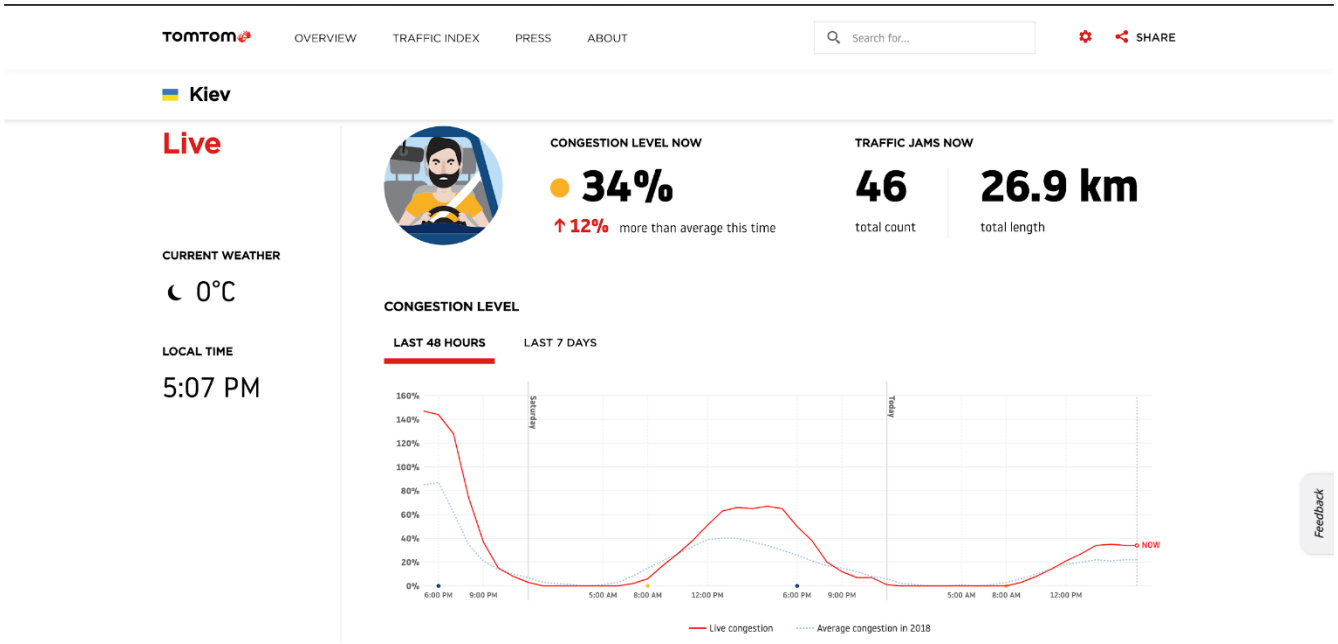


Рисунок 1.4. TomTom Traffic Index

Крім розробок приватних компаній, існують також ініціативи державних органів управління, що стосуються розумного міста та надання послуг в онлайн режимі. Зокрема, влада міста Відень за підтримки приватної компанії EY

розробила муніципальну платформу відкритих даних, яка використовує технології блокчейн-мереж для перевірки і захисту відкритих даних, таких як маршрути громадського транспорту, результати голосування громад, звіти енергоспоживання та реєстраційні дані підприємств. З моменту запуску проекту в грудні 2017 року вдалося захистити близько 350 масивів даних за допомогою блокчейн-мереж. Мережі, що були запущені одними з перших в Європі, захищають офіційні документи, зберігаючи хеші наборів даних у загальнодоступних блокчейн-мережах.

Проте, для того, аби отримати комплексний погляд на стан міста та застосувати системний підхід до вирішення проблем недостатньо реалізовувати проекти, що вирішують специфічну задачу певної області господарства. На зміну таким видам систем приходять горизонтальні платформи розумних міст [4]. Дані системи застосовують принципи Інтернету Речей. Це дає змогу керувати різними девайсами та інтегрувати різні джерела даних для покращення рівня життя жителів. На основі платформи будуються прикладні застосунки, з якими взаємодіють кінцеві користувачі. Іншими словами, платформа розумного міста виступає фреймворком для комунікації, інтеграції і інтелектуальної обробки даних [5].

Серед комерційних продуктів, які являють собою такі когнітивні фреймворки, IBM Watson пропонує систему з рядом аналітичних послуг, реалізація яких покладається на динамічне машинне навчання. Навчальний процес постійно вдосконалюється за рахунок аналізу результатів попередніх раундів. Як визначає IBM, когнітивні обчислення – це термін, що описує системи, які навчаються використовуючи широких набір даних, можуть взаємодіяти з людиною за допомогою методів розпізнавання мови та застосовувати здобуті знання відповідно до контексту, в якому знаходяться.

Google Асистент – це сервіс, який надає користувачам корисну інформацію залежно від контексту (часу та місця перебування). Ця система вдосконалюється аналізуючи минулу поведінку користувачів, їх взаємодію з обліковими записами Google: Google Calendar, Chrome, Gmail, Пошук та Youtube.

Haven OnDemand2 – це платформа для розробників від компанії HP Enterprise, яка використовує можливості машинного навчання та надає різні API для створення когнітивних сервісів, зокрема: засоби для аналізу тексту, розпізнавання мовлення, аналізу зображень, індексації та пошуку.

Серед наукових розробок також існує декілька робіт, які пропонують когнітивні рішення, що можуть задовольнити потреби систем, які базуються на використанні IoT технологій (зокрема, систем розумного міста). P. Vlacheas з колегами розробив фреймворк, який дозволяє комунікувати розумним об’єктам міста з найбільш релевантними об’єктами та, у результаті, приносити більшу цінність кінцевим користувачам [6]. У їх роботі дослідники зосередились на повторному використанні функціональності та послуг доступних об’єктів через три рівні, що включають в себе віртуальні об’єкти (VO), складені віртуальні об’єкти (CVO) та рівень обслуговування. Автори показали, що час надання послуг завдяки використанню фреймворку скорочується, що призводить до зниження операційних витрат.

Cognitive Internet of Things (CIoT) – ще одне дослідження, яке було проведено Q. Wu разом з його командою [7]. Розроблений фреймворк пропонує взаємодію між п’ятьма когнітивними задачами: цикл сприйняття-дії, аналітика великих даних, семантичне виведення та знаходження знань, інтелектуальне прийняття рішень, надання послуги за запитом. У роботі визначено два аспекти, необхідні для надання об’єктам когнітивного середовища ознак здатності до навчання та розуміння: розумні об’єкти повинні вміти виводити смисли з проаналізованих даних та виявляти цінні шаблони та закономірності.

Автори дослідження про фреймворк для розумних будинків на базі динамічних когнітивних систем та IoT [8] використали Байєсівську модель, Байєсівський фільтр та машинне навчання з підкріпленням (Reinforcement learning). Байєсівська модель розміщується поверх перцептора, який спостерігає за навколишнім середовищем. Байєсівський фільтр оцінює стан системи, а алгоритми машинного навчання з підкріпленням забезпечують механізм вибору найкращих можливих дій на основі загальної кількості отриманих винагород.

На відміну від централізованої системи прийняття рішень та аналітики, розгорнутої в хмарному середовищі, автори роботи «Incorporating Intelligence in Fog Computing for Big Data Analysis in Smart Cities» [9] запропонували інтегрувати штучний інтелект в туманні обчислення для полегшення аналізу великих даних розумного міста. Вони запровадили ієрархічну модель туманного обчислення для аналізу великих даних для інтелектуальних програм міст. Використовуючи цю модель, загальна продуктивність системи підвищується за рахунок зменшення об’ємів передачі даних, так як зникає необхідність передавати всі необроблені дані до сервера, який розгорнутий в хмарному середовищі. Крім цього, через близькість туманної інфраструктури до джерел даних, стає можливим виконання аналітики в режимі реального часу. У роботі також застосована прихована марковська модель (HMM) для підтримки аналізу великих даних.

У табл. 1.1 наведені короткі відомості розглянутих проектів та розробок в сфері горизонтальних платформ для розумного міста. Зокрема, показано рівні генерації великих даних, які аналізуються цими системами, алгоритми машинного навчання, що застосовуються та найбільш доцільні сфери використання.

Таблиця 1.1. Порівняння існуючих платформ для розумного міста

Існуючі рішення	Підтримка машинного навчання та аналізу			Домен	Алгоритми машинного навчання	Сфери використання
	Рівень інфраструктури	Рівень туманних обчислень	Рівень хмарних обчислень			
IBM Watson			✓	Загального використання	Різні	Охорона здоров’я, Розкриття злочинів

Таблиця 1.1. Закінчення

Google Асистент			✓	Націлений на користувачів	Різні	Керування автомобільним трафіком, переміщеннями
HPE Haven OnDemand			✓	Загального використання	Різні	Аналіз емоцій людей
Cognitive management framework	✓	✓		Розумне місто	Розпізнавання образів, методи семантичного обгрунтування	Охорона здоров’я, Громадська безпека, Розумний транспорт
Cognitive IoT			✓	Розумне місто	Мультиагентне навчання	Розумний будинок, Маршрутизація трафіку в реальному часі
Cognitive interactive framework		✓		Розумний будинок	Машинне навчання з підкріпленням	Розумний будинок
Intelligence in Fog		✓		Розумне місто	Прихована марковська модель	Автоматизація в розумному місті
Intelligent gateway		✓		IoT	Навчання на правилах (RBML)	Охорона здоров’я

1.4 Постановка задачі

Основною мотивацією створення системи розумного міста є потреба у вирішенні зростаючих проблем сучасних міст за допомогою використання передових технологій комунікації, обробки інформації та інтелектуального прийняття рішень. Планується спроектувати та розробити платформу, яка буде взаємодіяти з розумними об’єктами міста, зберігати інформацію, аналізувати її та знаходити корисні знання, шаблони та закономірності, які допоможуть

ефективніше планувати та керувати містом. На базі платформи буде створено модуль Керування автомобільним трафіком, який дозволить користувачам прокладати оптимальні маршрути в межах міста, таким чином, знижуючи навантаження на дороги та зменшуючи час поїздок. Для ефективного аналізу даних сенсорів, камер відеоспостереження та смартфонів користувачів сервісу використовуватимуться різні методи та алгоритми машинного навчання.

Розроблена система розумного міста працюватиме в середовищі з наступними характеристиками:

- сенсори та розумні пристрої міста генерують дані з великою швидкістю. Не вся вхідна інформація може бути переглянута і перевірена людиною на коректність, але система повинна вміти покращуватись аналізуючи попередній досвід;
- контекст платформи розумного міста не є чітким. Середовище в якому працює система змінюється з часом. Як наслідок, система повинна мати механізм динамічного та постійного навчання;
- дані, які генеруються об’єктами розумного міста часто можуть бути пошкодженими, містити неточності.

Зважаючи на вищезазначені особливості середовища, в якому буде розгорнута система, вирішено застосувати машинне навчання (глибинні нейронні мережі, навчання з підкріпленням, часткове машинне навчання) для вирішення задач домену.

Для досягнення мети необхідно вирішити цілий ряд задач, зокрема:

1. Аналіз доступних технологій для побудови платформи, що буде розгортатись в хмарному середовищі.
2. Проектування архітектури платформи: визначення основних компонентів системи, підбір інструментів потокової обробки даних, баз даних для зберігання неструктурованої інформації, інструментів аналітики та візуалізації даних, збору метрик платформи.
3. Розробка топології інфраструктури системи.

4. Аналіз доступних методів та алгоритмів машинного навчання; підбір алгоритмів машинного навчання для вирішення задачі знаходження оптимального маршруту в умовах міста.
5. Розробка моделі нейронної мережі модуля Керування автомобільним трафіком.
6. Навчання нейронної мережі за допомогою тестового набору даних.
7. Дослідження ефективності розробленої моделі пошуку оптимального маршруту.
8. Оптимізація нейронної мережі за результатами дослідження її ефективності.
9. Реалізація API для доступу до функціональності модуля Керування автомобільним трафіком.

Розробка системи розумного міста пов’язана з різними технічними викликами, які потрібно вирішити, для того, аби забезпечити надійність та коректність роботи системи. Одним з найбільших викликів є забезпечення безпеки даних та збереження конфіденційності чутливої інформації кінцевих користувачів. Підходи машинного навчання, які базуються на використанні великої кількості даних (наприклад, методи глибинного машинного навчання) можуть бути уражені атаками зловмисників, зокрема ін’єкцією пошкоджених даних (англійською False Data Injection). Це може компрометувати систему. Стійкість проти таких атак є обов’язковою вимогою до алгоритмів машинного навчання, які будуть використовуватись в платформі. Збереження конфіденційності – це важливий фактор, так як значна кількість інформації, що обробляється в системі, належить жителям, які повинні бути впевнені в тому, що їхня персональна інформація не є відкритою для інших осіб. Для прикладу, модуль Керування автомобільним трафіком повинен унеможливити сценарії несанкціонованого доступу до даних місцезнаходження користувачів сервісу. Без вирішення цих проблем широке впровадження системи розумного міста є неможливим.

Інтеграція аналітики великих даних та швидкої аналітики. В контексті розумного міста існує багато застосунків, які є чутливими до часу та потребують аналітики потоку даних в режимі реального часу (наприклад, розумний транспорт, самокеровані автомобілі). Такі рішення потребують нових інструментів аналітики, фреймворків, що підтримують аналітику великих даних в поєднанні з методами швидкого аналізу.

Інтелектуальність об’єктів розумного міста. Розглядається можливість використання спрощених алгоритмів машинного навчання, розгорнутих на пристроях з обмеженими обчислювальними ресурсами, для складних задач інтелектуального прийняття рішень в режимі реального часу. Це також частково задовольняє вимогу щодо безпеки та захисту приватної інформації, так як в цьому випадку дані не будуть передаватись через канали зв’язку до серверних застосунків, які розгорнуті в хмарному середовищі.

Дефіцит масивів тестових даних. При розробці та оцінці ефективності нейронних мереж для системи розумного міста потрібно зібрати великі набори даних, що відображають реальні процеси в місті. Це завдання є дуже трудомістким для більшості сфер господарства.

Усвідомлення контексту. Для того, аби отримати більше цінності від вхідних даних та прийняти більш раціональне рішення, нейронні мережі, які розробляються, повинні розглядати вхідні сирі дані в контексті домену, завдання якого вирішує програма. Для прикладу, система визначення стану людини повинна діяти зовсім іншим чином в контекстах водіння автомобіля та відпочинку вдома при класифікації сонного обличчя у людини.

Крім вищезазначених викликів, існують також інші проблеми, які впливають на дизайн системи розумного міста: інтеграція різних фреймворків для аналітики, розподіл операцій аналізу та прогнозування, відсутність комплексного середовища для повного тестування розроблених рішень.

1.4.1 Функціональні вимоги

Функціональні вимоги до програмного забезпечення описують роботу системи та її окремих компонентів. Представляють собою опис особливостей обробки даних, її трансформації, очікувану поведінку на дії користувачів чи запити сторонніх сервісів, з якими взаємодіє система, що розглядається. Зазвичай, функціональні вимоги виражають в формі «система повинна вміти виконувати...» та визначаються наприкінці першої стадії процесу розробки програмних застосунків – на етапі аналізу вимог.

Проаналізувавши предметну область системи, що розробляється, концепцію розумного міста та потреби жителів сучасних метрополісів, розглянувши та порівнявши наявні рішення, було сформовано наступні функціональні вимоги до системи:

1. Система повинна вміти визначати місцезнаходження користувача на початку сеансу взаємодії користувача з програмним застосунком, місцезнаходження локацій, які користувач шукає.
2. Система повинна вміти відображати поточний стан автомобільного трафіку в місті, рівень заторів та середню швидкість руху транспорту на окремих ділянках вулиць.
3. Система повинна вміти враховувати повідомлення користувачів про аварії та інші події на дорозі при оцінці стану трафіку та розрахунку маршрутів.
4. Система повинна вміти розраховувати оптимальний маршрут: від місцезнаходження користувача до місця призначення чи від довільної локації в місті до місця призначення та надавати результат в зручному для користувачів вигляді: з переліком вулиць, очікуваним часом прибуття тощо.
5. Система повинна зберігати прокладені маршрути користувачів та надавати зручний доступ до історії пересувань, можливість переглянути деталі поїздки (час пересування та інші особливості).

6. Система повинна надавати детальну статистику стану трафіку: рівень заторів для різних періодів дня, часи пікових навантажень транспортної системи, середній додатковий час поїздки у випадку наявності інтенсивного трафіку.
7. Система повинна надавати можливість користувачам спілкуватись в чатах, які прив’язані до локації, в якій в даний момент знаходяться користувачі. Це дозволить зменшити стрес у випадку високого рівня заторів.
8. Система повинна надавати можливість планувати майбутні поїздки. При настанні часу запланованої поїздки сповіщати користувача про необхідність початку подорожі.
9. Система повинна надавати можливість користувачам ділитись своїм місцезнаходженням, маршрутом, статусом та очікуваним часом прибуття поточної поїздки.
10. Система повинна вміти створювати облікові записи користувачів, надавати інструменти керування обліковим записом: зміна пароля чи загальної інформації користувача, налаштування сповіщень.

1.4.2 Нефункціональні вимоги

Нефункціональні вимоги описують обмеження та залежності системи, що розробляється. Вони доповнюють функціональні вимоги, які визначають поведінку системи, та описують якою саме повинна бути система з погляду безпеки, надійності, швидкодії, зручності у використанні, відновлюваності. Зазвичай нефункціональні вимоги виражаються у формі «система повинна бути...».

Разом з функціональними вимогами були визначені наступні нефункціональні вимоги:

1. Система повинна бути доступною більше 99,9% часу роботи. Тобто, за класифікацією відноситись до систем високої доступності. Для

досягнення цієї вимоги потрібно розробити стратегію зменшення та управління збоями, мінімізувати час планових простоїв. Реалізація цієї вимоги є критичною, адже недоступність сервісу призведе до значних економічних втрат міста та зростання вартості перевезень товарів та людей.

Для визначення досягнутого рівня доступності можна використовувати два метода: відсотковий метод та метод напрацювання на відмову. Відсотковий метод дозволяє визначити досягнутий рівень доступності використовуючи значення часу непередбаченого простоювання та часу обіцяної доступності, який не включає всі заплановані періоди недоступності (при проведенні оновлення системи чи діагностики). Це можна виразити формулою 1.1:

$$доступність = (Д — П) / Д \times 100\ \%, \quad (1.1)$$

де: *Д* – час обіцяної доступності, *П* – час непередбаченого простоювання. Таким чином, для системи з заявленою доступністю 99,9% 40 хвилин в місяць є максимальним дозволеним часом простоювання.

Доступність також можна виразити через середні величини (формула 1.2):

$$середня\ доступність = MTTF / (MTTF + MTTR) \times 100\ \%, \quad (1.2)$$

де: *MTTF* (Mean time to failure) – середнє напрацювання до відмови, *MTTR* (Mean time to repair) – середній час до відновлення працездатності. Час відновлення після збою залежить від багатьох факторів, таких як складність системи (чим складніше система, тим довше триває її перезапуск), серйозність проблеми, доступність обслуговуючого персоналу, запасного обладнання. Слід також зазначити, що доступність

системи вимірюється з точки зору користувача, а не підтвердження факту роботи основних вузлів.

2. Система повинна надійно захищати персональні дані користувачів. Платформа розумного міста, а також її модулі, зокрема модуль Керування автомобільним трафіком будуть обробляти велику кількість чутливої інформації користувачів. Потрібно забезпечити надійні канали зв’язку між компонентами системи, використовувати протоколи передачі даних, які здійснюють шифрування (наприклад, HTTPS).
3. Система повинна надавати зручний API для розробників прикладних програмних застосунків. Модуль Керування автомобільним трафіком буде займатись аналізом поточного стану завантаження доріг та обчисленням оптимального маршруту за допомогою використання нейронних мереж та алгоритмів машинного навчання. Відображення цієї інформації буде відбуватись в мобільних додатках та веб-сайтах. Для того, аби сторонні розробники могли ефективно розвивати свої додатки, API розроблюваної системи повинен бути стандартизований, інтуїтивно зрозумілий та простий у використанні.
4. Система повинна бути швидкою. Більшість задач, які вирішує система, повинні обробляти нову інформацію в режимі реального часу та миттєво сповіщати користувача. Наприклад, аварії на дорогах чи затори можуть впливати на маршрут, який будується модулем Керування автомобільним трафіком.
5. Система повинна бути універсальною. Розроблювана платформа розумного міста – це комплексна система, яка покликана вирішувати проблеми різних сфер господарства та інтегрувати різні пристрої, датчики та сенсори.

Висновки по розділу

У результаті аналізу предметної області визначено основні проблеми міст, які розроблювана система повинна вирішити. Досліджено концепцію розумного міста, основні принципи розробки систем для автоматизації процесів міста.

Проведено детальний аналіз та порівняння наявних рішень, визначено їхні сфери застосування та обмеження. Доведено актуальність системи, яка розробляється.

Сформульовано об’єкт на предмет дослідження, призначення системи та її практичну цінність. Для модуля Керування автомобільним трафіком визначено сукупність функціональних та нефункціональних вимог.

РОЗДІЛ 2. АНАЛІЗ ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Цей розділ описує технології, які використовуються в системі розумного міста. Завдяки використанню технологій Інтернету Речей розроблена платформа здатна об’єднати різні джерела даних та отримати комплексний погляд на процеси міста, застосувати системний підхід до вирішення проблем. Для обробки великої кількості інформації, які генерують сенсори та розумні пристрої, використано платформу потокової обробки даних. Інтелектуальний аналіз та прогнозування, здатність адаптуватись до мінливого навколишнього середовища є можливими завдяки застосуванню в системі алгоритмів машинного навчання, нейронних мереж. Для того, аби система могла зберігати величезні об’єми даних, були обрані різні noSQL бази даних відповідно до задач, які вони здатні ефективно вирішувати. Розгортання компонентів системи в хмарному середовищі дозволило системі легко масштабуватись та задовольняти пікові навантаження, зменшити час від розробки нової функціональності до її використання кінцевими користувачами. Провайдери хмарних технологій надають потужні інструменти аналітики та моніторингу стану сервісів, баз даних та інших інфраструктурних компонентів.

2.1 Технології Інтернету Речей

Інтернет речей (IoT) – це система взаємопов'язаних обчислювальних пристроїв, датчиків і сенсорів, яким надаються унікальні ідентифікатори та можливість передавати дані по мережі без потреби взаємодії типу людина-людина або людина-комп'ютер.

В концепції технології IoT «річчю» може називатись людина з імплантом для моніторингу стану серця, сільськогосподарська тварина з розумним біочіпом, автомобіль, який має вбудовані датчики тиску в шинах, сенсори відкриття дверей та відеокамери, або будь-який інший об'єкт, якому може бути призначена IP-адреса і який здатний передавати дані по мережі.

Все частіше організації в різних галузях використовують IoT для більш ефективної роботи, покращення рівня обслуговування клієнтів, прийняття рішень та підвищення цінності бізнесу.

Технологія IoT розвинулась завдяки конвергенції бездротових технологій, мікроелектромеханічних систем (MEMS) та Інтернету. Це призвело до об’єднання операційних технологій (OT) з інформаційними технологіями (IT) і, як наслідок, дозволило аналізувати неструктуровані дані, що генеруються пристроями, та виявляти цінні інсайти. Інтернет Речей розвиває ідею міжмашинної комунікації (M2M communication), яка полягає в обміні даних між пристроями через мережу без участі людини.

Інтернет речей має спільні риси з системами SCADA (supervisory control and data acquisition), категорії програмного забезпечення для управління процесами, збору даних у режимі реального часу з віддалених місць для контролю обладнання та автоматизації процесів. Системи SCADA включають апаратні та програмні компоненти. Апаратне забезпечення збирає та передає дані до комп’ютера, на якому встановлено програмне забезпечення SCADA, де вони потім обробляються. Еволюція систем SCADA відбулась таким чином, що системи SCADA пізнього покоління являються IoT системами першого покоління.

IoT екосистема складається з розумних пристроїв, які використовують вбудовані процесори, датчики та комунікаційне обладнання для збору, надсилання та реагування на дані, які вони отримують із свого середовища. Об’єкти Інтернету Речей діляться даними, які вони збирають, підключаючись до IoT шлюзу (IoT Gateway), де дані надсилаються до хмари для аналізу. Іноді ці пристрої комунікують з іншими пов'язаними пристроями та діють відповідно до даних, які вони отримують один від одного. Пристрої виконують більшу частину роботи без втручання людини, хоча люди можуть взаємодіяти з ними, наприклад, щоб налаштувати їх або отримати доступ до даних.

Протоколи підключення, мережевих зв’язків та комунікацій, що використовуються цими веб-пристроями, значною мірою залежать від конкретних сфер застосування та використовуваних платформ.

На рис. 2.1 зображена схема взаємодії компонентів IoT системи та їх призначення.

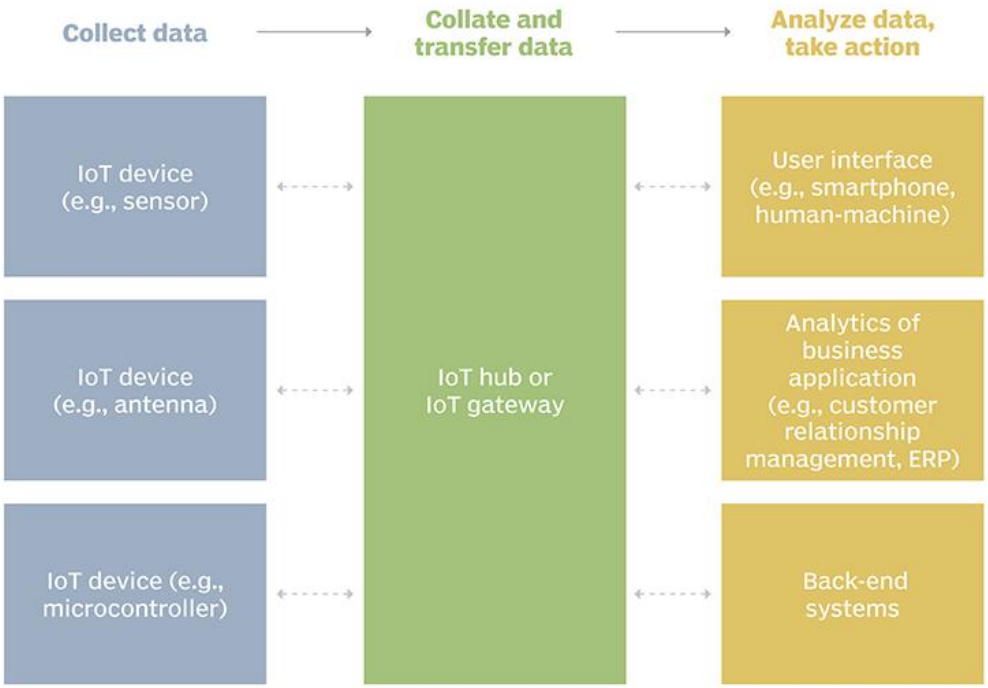


Рисунок 2.1. Приклад IoT системи

Як і будь-яка інша технологія, Інтернет Речей має сильні сторони та обмеження, які варто розглянути при проектуванні та розробці системи розумного міста.

До сильних сторін технології відносяться:

- автоматизація завдань допомагає покращити якість послуг бізнесу та зменшує потребу у втручанні людини;
- можливість доступу до інформації з будь-якого місця в будь-який час на будь-якому пристрої;
- передача пакетів даних через підключену мережу економить час та гроші;
- поліпшення зв'язку між підключеними електронними пристроями.

Разом з цим, на сьогоднішній день, технології IoT мають певні обмеження та складнощі в імплементації:

- у випадку, коли підприємства мають справу з величезною кількістю IoT пристроїв, збір, управління та аналіз даних з усіх пристроїв представляє собою складну задачу;
- зі збільшенням кількості підключених пристроїв та обсягів інформації, яка передається між пристроями, збільшується й імовірність того, що зловмисники можуть вкрасти конфіденційну інформацію;
- досить важко налаштувати комунікацію між пристроями різних виробників, так як досі існує проблема в стандартизації протоколів міжмашинної комунікації;
- якщо в платформу вносяться зміни, що містять дефект, підключені пристрої можуть почати працювати некоректно.

Екосистема Інтернету Речей характеризується великим набором доступних технологій комунікації між розумними пристроями, операційними системами для девайсів з обмеженими обчислювальними ресурсами, мережевими протоколами та комплексними фреймворками, які надають провайдери хмарної інфраструктури. Нижче наведені найбільш популярні стандарти в сфері IoT.

OneM2M – сервісне рішення для міжмашинної комунікації, яке можна інтегрувати в програмне та апаратне забезпечення для підключення пристроїв. Технологія була створена для розробки стандартів, які дають можливість IoT додаткам взаємодіяти один з одним.

6LoWPAN (IPv6 на базі бездротових персональних мереж з низькою потужністю) – це відкритий стандарт, розроблений організацією Internet Engineering Task Force (IETF). Стандарт 6LoWPAN дає змогу будь-яким пристроям, що використовують радіохвилі малої потужності комунікувати через мережу Інтернет, включаючи пристрої, що працюють з 804.15.4, Bluetooth Low Energy та Z-Wave.

ZigBee – бездротова мережа з доволі низькою швидкістю передачі даних та ефективним використанням енергії, що використовується переважно для промислових цілей. ZigBee розроблений на базі стандарту IEEE 802.15.4. Організація, яка створила технологію ZigBee також представила Dotdot – універсальну мову для IoT систем, яка дозволяє розумним об’єктам комунікувати один з одним в будь-якій мережі надійно та захищено.

DDS (Data Distribution Service) – служба розподілу даних, яка була розроблена організацією OMG (Object Management Group) та є стандартом IoT для ефективної та масштабованої комунікації в режимі реального часу.

LiteOS – Unix-подібна операційна система для пристроїв з обмеженими обчислювальними ресурсами. LiteOS підтримує смартфони, електронні часи, інтелектуальні датчики на підприємствах та пристрої розумного будинку.

AMQP (Advanced Message Queuing Protocol) – рішення з відкритим кодом для асинхронної доставки повідомлень. AMQP дозволяє додаткам та організаціям обмінюватись повідомленнями через надійні канали, які забезпечують шифрування інформації. Протокол використовується для комунікації клієнт-серверних систем та керування IoT пристроями.

LoRaWAN (Long Range Wide Area Network) – протокол, розроблений для підтримки великих мереж, таких як мережа розумного міста, що може об’єднувати мільйони пристроїв малої потужності.

CoAP (Constrained Application Protocol) – протокол, розроблений організацією IETF, для комунікації обчислюваних пристроїв з обмеженою потужністю в середовищі IoT.

Також існують фреймворки з відкритим кодом та комерційні IoT платформи, які дозволяють швидко та ефективно вирішити стандартні задачі IoT системи.

AWS IoT – IoT платформа, розроблена хмарним провайдером - компанією Amazon. Створена для того, щоб розумні пристрої могли легко підключатися та безпечно взаємодіяти з хмарними сервісами AWS та іншими підключеними пристроями.

Microsoft Azure IoT Suite – платформа, що складається з набору сервісів, які дозволяють користувачам взаємодіяти та отримувати дані зі своїх IoT пристроїв, а також виконувати різні операції над даними, зокрема: багатовимірний аналіз, трансформація та агрегація даних, візуалізація даних у спосіб, який допоможе бізнесу розуміти природу даних системи.

Brillo/Weave – платформа від компанії Google для швидкої імплементації IoT застосунків. Платформа складається з двох основних систем: Brillo, операційної системи на основі системи Android для розробки вбудованих пристроїв малої потужності, та Weave, протоколу передачі даних в IoT середовищі, який служить засобом зв'язку між девайсами та хмарою.

ARM Mbed IoT – платформа для розробки IoT додатків на базі ARM мікроконтролерів. Головна задача платформи – створити масштабоване та безпечне середовище для пристроїв шляхом інтеграції Mbed інструментів та сервсів.

Calvin – IoT платформа з відкритим сирцевим кодом, випущена компанією Ericsson. Призначена для створення та управління розподіленими програмами, що дозволяють пристроям комунікувати один з одним. Calvin включає фреймворк для розробки IoT додатків, а також середовище виконання для запуску застосунків.

2.2 Машинне навчання у контексті систем розумного міста

Розвиток систем розумних міст призводить до генерації великої кількості даних з безпрецедентними темпами. На жаль, більша частина інформації марно втрачається без вилучення потенційно корисних знань через відсутність встановлених механізмів, які можуть ці дані аналізувати. Така динамічна природа середовища сучасних міст вимагає появи нового покоління підходів до машинного навчання, які є достатньо гнучкими, щоб впоратися з мінливою природою даних розумного міста.

Однією з найбільших проблем сучасних міст у контексті машинного навчання є відсутність достатньої кількості мічених даних. Це створює потребу у використанні алгоритмів машинного навчання, які здатні ефективно використовувати та поєднувати як мічені набори даних, так і сирі дані.

Незважаючи на останні досягнення в галузі обчислювальної техніки та технологій зберігання даних, більшість підходів аналітики даних використовують методи вибірки, які є ефективними з погляду часу виконання, проте можуть не врахувати велику частину даних, які можуть містити цінні знання та закономірності, що не представлені вибірками. З іншого боку, завдяки використанню глибинних нейронних мереж (DNN), набори даних з мільйонами параметрів можна ефективно використовувати для отримання цінних інсайтів.

Необхідність використання глибинних нейронних мереж виникає з необхідності вилучення абстракцій високого рівня з необробленої інформації. Кожен шар нейронної мережі створює абстрактну репрезентацію вхідних даних. Щоб отримати більше рівнів абстракції, потрібно додати більше прихованих шарів нейронів.

Ще одним видом машинного навчання, яке успішно може бути застосоване в області розумних міст є навчання з підкріпленням (RL – Reinforcement Learning). Цей вид навчання добре показався в системах управління та автоматизації рутинних задач. Особливістю навчання з підкріпленням є те, що для тестових даних не визначається правильний результат (тобто, не вирішується проблема класифікації) – це характерно для багатьох завдань систем розумного міста – натомість, винагороджується вибір правильної дії. Кінцева мета системи, яка застосовує алгоритми RL – знайти дію для кожного стану системи таким чином, щоб загальна винагорода навчального агента була максимальною в довгостроковій перспективі. Проте, з іншого боку, неможливо надати відгук про всі дії, які обрала система навчаючись на наборі тестових даних. Це питання можна вирішити завдяки використанню часткового машинного навчання (semi-supervised learning), де набір навчальних даних частково позначений.

Підходи часткового (напівавтоматичного) машинного навчання є перспективними методами вирішення проблеми дефіциту анотованих даних у великих потоках даних. Більше того, глибинне навчання з підкріпленням (DRL – Deep Reinforcement Learning) також показали багатообіцяючі результати в системах, де необхідний зворотний зв'язок із навколишнього середовища для підвищення продуктивності системи. Поєднання цих методів може допомогти отримати більшу цінність із великої кількості даних, які генеруються об'єктами розумного міста.

У системі розумного міста, яка розробляється, поєднані ці підходи для створення моделі, яка здатна навчатись на основі даних розумного міста та змінювати навколишнє середовище найкращим чином. Розроблене рішення здатне розвиватись та адаптуватись до зміни умов зовнішнього середовища та виконувати самостійні дії без втручання людини.

2.2.1 Особливості глибинного навчання з підкріпленням

Навчання з підкріпленням має на меті імітувати процес навчання людей. За допомогою цього методу навчання агент може сприймати навколишнє середовище використовуючи різні джерела даних. Агент використовує джерела вхідних даних для узагальнення досвіду роботи системи для протистояння новим та невідомим ситуаціям. Поєднання навчання з підкріпленням та глибинних нейронних мереж – відоме як навчання з глибинним підкріпленням DRL – вирішує декілька обмежень RL методів, включаючи обмеження у областях застосування та їх погані характеристики масштабування у контексті багатовимірних доменів.

DRL модель спостерігає за параметрами навколишнього середовища, здійснює дії щодо навколишнього середовища та отримує зворотній зв'язок у вигляді нагороди за кожну дію. Мета агента – максимізувати значення винагород. Глибинна нейронна мережа (DNN) використовується для апроксимації оптимальної функції дії-значення (іншими словами, яку дію

найкраще здійснити при даному стані системи, щоб отримати максимальне значення винагороди у майбутньому). Рис. 2.2 ілюструє концептуальну структуру DRL системи.

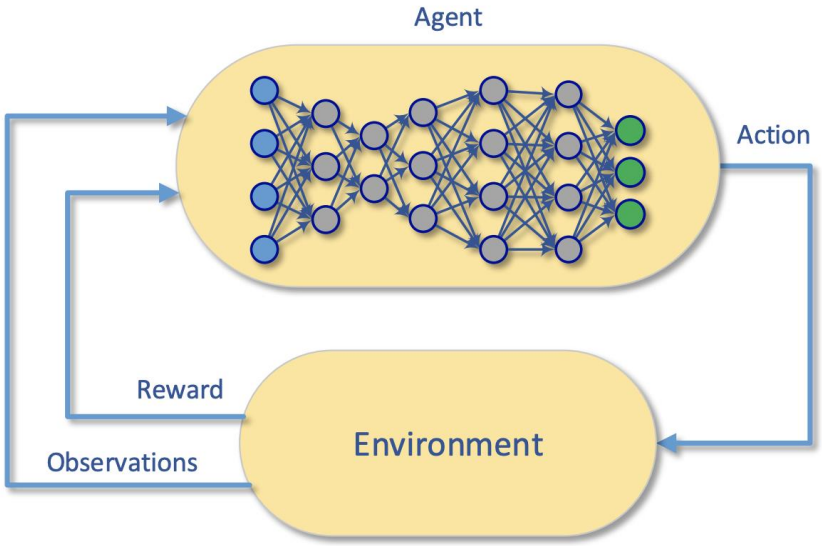


Рисунок 2. 2. Концептуальна структура глибинної системи з підкріпленням

В розроблюваній системі розумного міста модель машинного навчання використовує генеративні глибинні нейронні мережі як компонент часткового навчання. Модель визначає вірогідність виконання кожної дії в системі.

На рис. 2. 3 порівнюються DRL моделі контрольованого та часткового машинного навчання для тестової системи розумного кампусу – проекту в контексті домену розумного міста [11]. Головним завданням проекту є надання послуг локалізації та навігації у приміщеннях кампусу. Функція винагороди розробленої моделі еквівалентна відстані до точки призначення. Модель вчиться на основі даних відбитків пальців RSSI зчитувачів, щоб вжити найкращих дій (тобто, визначити рух до місця призначення). Результати показують, що використання DRL моделі часткового навчання, яка застосовує комбінацію анотованих та сирих даних покращує продуктивність. З точки зору здобуття більшого значення винагород, модель часткового навчання швидше починає отримувати вищі нагороди порівняно з моделлю контрольованого навчання.

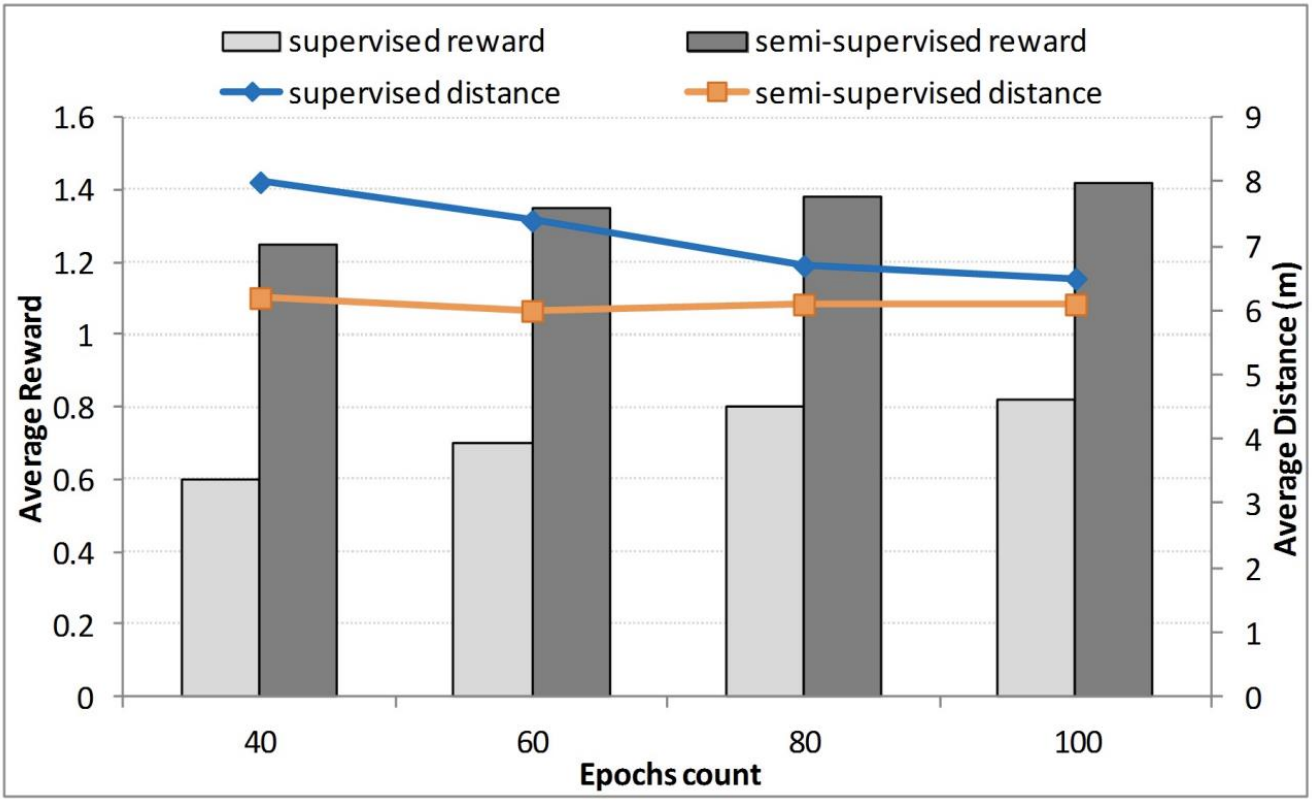


Рисунок 2. 3. Ефективність DRL моделі часткового навчання порівняно з DRL моделлю керованого навчання

Вона отримує від 60% до 100% більше винагород в порівнянні з моделлю керованого навчання в різних моментах часу. Щодо точності локалізації, модель часткового навчання наближається до очікуваного результату, досягаючи покращення у 6 - 23% порівняно з точністю моделі контрольованого навчання.

Розглянута DRL модель часткового навчання може використовуватись в платформі розумного міста на рівні хмарних та туманних обчислень, оскільки глибинна нейронна мережа, що лежить в основі моделі, є складною та ресурсо-затратною, тому не може розміщуватися на IoT пристроях, які мають обмежені обчислювальні ресурси. Потрібно провести більше досліджень, щоб адаптувати алгоритм до обмежень існуючих IoT пристроїв.

2.3 Потокова обробка даних

Основними джерелами даних системи розумного міста, що розробляється, є численні датчики, сенсори та розумні пристрої. На основі отриманої інформації система приймає рішення та здійснює дії щодо навколишнього середовища. Наприклад, інформація датчиків визначення швидкості руху транспортних засобів, камер відеоспостереження та даних місцезнаходження користувачів використовується модулем Керування автомобільним трафіком для побудови маршрутів та відображення актуального стану рівня заторів у місті. Розумні об’єкти взаємодіють з серверною частиною платформи за допомогою IoT шлюзу (IoT Gateway), який здійснює фільтрацію та подальну відправку даних у відповідні бекенд системи. Очікується, що система буде отримувати величезну кількість інформації, яку неможливо коректно обробляти без застосування проміжних платформ потокової обробки повідомлень.

Apache Kafka – це розподілена система, яка забезпечує надлишковість для зберігання величезних об’ємів даних. Вона надає шину для відправки повідомлень з великою пропускною здатністю, завдяки якій можна обробляти вхідні дані в режимі реального часу. За іншим визначенням, Kafka – це розподілений горизонтально масштабований відмовостійкий журнал комітів.

Розподіленість Kafka полягає в особливості зберігання, отримання та відправки повідомлень, які організовані на різних вузлах кластера. Кластер – сукупність машин, на яких розгортається система. Для кінцевих користувачів кластер Kafka виглядає як єдиний сервіс, що спрощує взаємодію з системою. Перевагами розподіленості є висока доступність та стійкість до відмов окремих компонентів платформи.

Стійкість до відмов. Як зазначалось вище, платформа Kafka є розподіленою. Це дозволяє уникнути ситуації, коли система має єдину точку відмови. Для прикладу, кластер Kafka з 5 вузлів залишається працездатним, навіть якщо 2 вузла припинили свою роботу через несправності.

Kafka підтримує горизонтальне масштабування. Це дозволяє розгортати платформу в хмарному середовищі, а у випадку підвищення навантаження на систему збільшувати кількість віртуальних машин. Горизонтальне масштабування в більшості випадків є кращим варіантом, ніж вертикальне масштабування, яке полягає в нарощуванні обчислювальних ресурсів існуючого сервера.

Для надійного зберігання вхідних повідомлень Kafka використовує структуру даних, яка гарантує порядок елементів – журнал комітів (транзакцій). Записи журналу комітів не можна ні видаляти, ні змінювати; інформація зчитується в одному напрямку – починаючи від найстаріших повідомлень. Схематично це зображено на рис. 2.4.

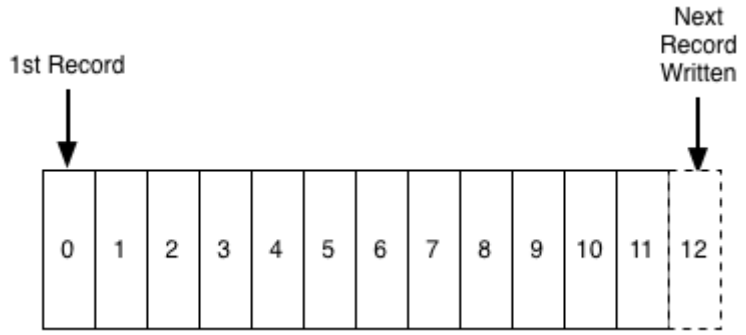


Рисунок 2. 4. Схема журналу транзакцій

Всі повідомлення, які надходять, зберігаються на диску та зчитуються послідовно завдяки особливостям вищезазначеної структури даних. Операції зчитування та запису виконуються за константний час O(1) та не впливають один на одного. Такі особливості реалізації системи дозволяють однаково ефективно та швидко працювати як з 100 Кб, так і з 100 Тб даних.

При роботі з Kafka виділяють додатки-продюсери та додатки-споживачі повідомлень. Додатки-продюсери надсилають повідомлення у визначений топік, який знаходиться на вузлі системи (брокері), а додатки-споживачі підписуються на отримання повідомлень топіка. Для кращого масштабування топіки розділяються на секції, які зберігають повідомлення у порядку їй надходження від додатків-продюсерів. Кожне повідомлення можна знайти за його зміщенням,

яке можна представити у вигляді звичайного індексу в масиві або порядковим номером, який збільшується на одиницю для кожного нового повідомлення у вибраній секції. Графічно будову топіка зображено на рис. 2.5.

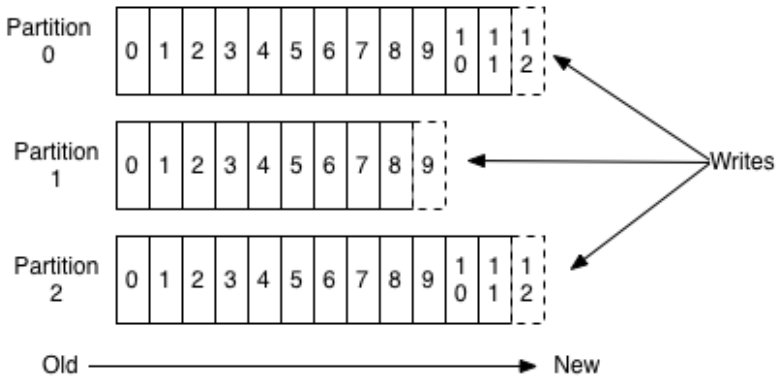


Рисунок 2. 5. Kafka Topic

Kafka не видаляє записи топіків після того, як додатки-споживачі їх обробили. Записи зберігаються протягом заданого періоду часу (наприклад, доби) або до тих пір, поки не буде досягнутий певний поріг використаної пам’яті. Додатки-споживачі самі опитують Kafka, чи не з'явилося у нього нових повідомлень, і вказують, які записи їм потрібно прочитати. Таким чином, вони можуть збільшувати або зменшувати зміщення, переходячи до потрібного запису; при цьому збережені повідомлення можуть повторно оброблятися.

Для швидшої обробки вхідних даних, додатки-споживачі об’єднують у групи (Consumer Group). Kafka гарантує, що декілька додатків, які відносяться до однієї групи, не можуть обробити одне й те ж повідомлення секції топіку, на який підписалась група. Іншими словами, секції топіку прив’язуються до одного додатка-споживача групи. Приклад потоку повідомлень зображено на рис. 2.6.

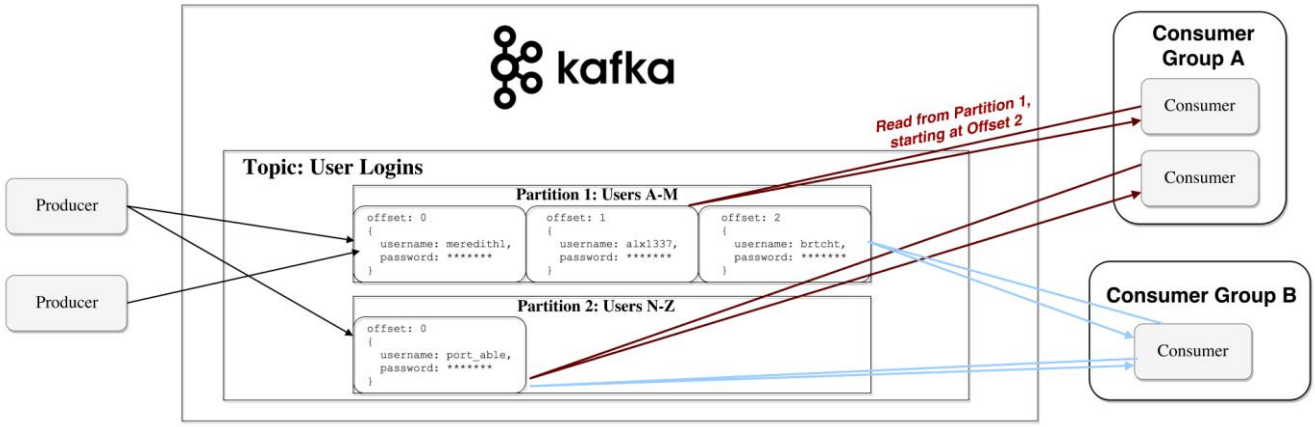


Рисунок 2. 6. Обробка повідомлень про логіни користувачів різними групами додатків-споживачів

2.4 Зберігання та аналіз неструктурованих даних

Система, що розробляється, отримуватиме велику кількість інформації від розумних об’єктів міста. При проектуванні системи важливо вибрати надійну технологію для зберігання неструктурованих даних та їх подальшого аналізу.

Apache Hadoop – це рішення з відкритим кодом для зберігання та обробки великих об’ємів даних, які зберігаються на різних вузлах кластера. Основними компонентами платформи є:

- розподілена файлова система HDFS (Hadoop Distributed File System);
- фреймворк та прикладний програмний інтерфейс для розробки MapReduce сервісів;
- Hadoop YARN – платформа для управління ресурсами, яка займається керуванням обчислювальних ресурсів кластерів;
- Hadoop Common – модуль, який містить бібліотеки, що використовуються іншими системами Hadoop.

Hadoop розбиває файли на великі блоки та розподіляє їх по вузлах кластера. Для запуску обчислювальних процесів упакований код передається у вузли для паралельного виконання. Цей підхід дозволяє використати перевагу локальності даних: вузли маніпулюють даними, до яких вони мають доступ. Це

дозволяє обробляти дані швидше та ефективніше, ніж це було б у класичних системах, які використовують паралельну файлову систему, де обчислення та дані поширюються за допомогою швидкісних мереж.

Всі модулі Hadoop розроблені з урахуванням можливості виникнення несправностей та збоїв в апаратному забезпеченні (окремих машин або цілих стелажів машин), а тому повинні автоматично оброблятися платформою. Компоненти MapReduce та HDFS Apache Hadoop розвинулись з дослідницьких робіт Google MapReduce та Google File System (GFS).

Крім HDFS, YARN та MapReduce, платформа Apache Hadoop складається з ряду пов'язаних проектів: Apache Pig, Apache Hive, Apache HBase та інших.

HDFS – це розподілена масштабована файлова система, написана на мові Java. HDFS кластер складається з наступних компонентів:

- Namenode – мастер-вузол, який займається управлінням файловою системою та зберігає метадані кластера: розміщення блоків файлів на вузлах даних, інформацію про реплікацію даних тощо. У випадку, якщо у кластері не налаштована опція бекапу Namenode вузла, вузол представляє собою єдину точку відмови.
- Datanode – вузол, який зберігає дані системи.
- Secondary Name Node – вузол, який зберігає контрольні точки метаданих файлової системи. Допоміжний вузол, який використовується для перезапуску Namenode мастер-вузла.

Файлова система HDFS розроблена для додатків, які не вимагають виконання операцій у режимі реального часу. Наприклад, файли не можуть бути модифіковані після збереження на диску, а затримка операцій читання/запису є досить великою, порівнюючи характеристики інших файлових систем. Проте, з іншого боку, пропускна здатність масштабується лінійно з додаванням нових вузлів, таким чином система може обробляти навантаження, яке у випадку використання одного сервера обробити було б неможливо. Дані, які зберігаються у файловій системі, можна отримати за допомогою використання RPC протоколу або CLI утиліти.

MapReduce Engine – фреймворк, який дозволяє запускати MapReduce обчислювальні процеси. Основними компонентами модуля є JobTracker, який бере на виконання задачі, займається плануванням, а також TaskTracker, який виконує задачі. За допомогою файлової системи JobTracker знає, який вузол містить потрібні дані, які машини знаходяться поблизу. Якщо обчислювальний процес не може бути виконаним на вузлі, де знаходяться дані, JobTracker запускає його на сусідній вузлах. Це дозволяє зменшити трафік у мережі. Якщо при виконанні процесу TaskTracker виходить з ладу, JobTracker перезапускає його і слідкує за станом виконання. Вузли, на яких запущений TaskTracker відправляють повідомлення про стан виконання процесу кожні кілька хвилин до JobTracker.

Висновки по розділу

В даному розділі проведено аналіз використаних технологій. Описано як вибрані технології вирішують задачі платформи розумного міста. Наведено особливості їх роботи, переваги та існуючі обмеження. Детально описано можливості технологій Інтернету Речей для збору інформації міста, способи ефективної передачі даних. Обґрунтований вибір моделі машинного навчання, яка використовує глибинні нейронні мережі та часткове навчання для ефективного вирішення задач в умовах постійної зміни середовища міста.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ

3.1 Архітектура системи

Система розумного міста, яка розробляється, на рівні інфраструктури складається з трьох рівнів, які в тій чи іншій мірі застосовують методи машинного навчання для вирішення задач: рівень IoT інфраструктури розумного міста, рівень туманних обчислень та рівень хмарних обчислень. На рис. 3.1 зображено підходи машинного навчання в межах ієрархії інфраструктури розумного міста, де кожен компонент системи контролюється інтелектуальним програмним агентом, який розгортається в туманному або хмарному середовищі залежно від характеристик завдання, яке вирішується (необхідність виконання в режимі реального часу). Наприклад, необроблені дані з розумних пристроїв можуть бути надіслані програмним агентам, які розгорнуті в туманному або хмарному середовищі. Запущений агент повертає відповідні дії на виконання пристроями після аналізу даних та прийняття рішення (у випадку з модулем Керування автомобільним трафіком це може стосуватись регулювання роботи світлофорів на основі даних про затори на відповідних дорогах).

Причина такої архітектури полягає в необхідності отримання різних рівнів абстракції даних стану процесів міста. На найвищому рівні (рівні хмарної інфраструктури) здійснюється загальноміське управління ресурсами та послугами міста на довгостроковій основі. З іншого боку, на найнижчому рівні, дані, які генеруються датчиками та розумними об'єктами, використовуються для управління ресурсами та послугами на короткостроковій основі. Крім цього, програмні агенти на рівні туманної інфраструктури приймають локальні рішення у заздалегідь визначеному контексті, тоді як хмарні обчислення стосуються великих географічних районів.

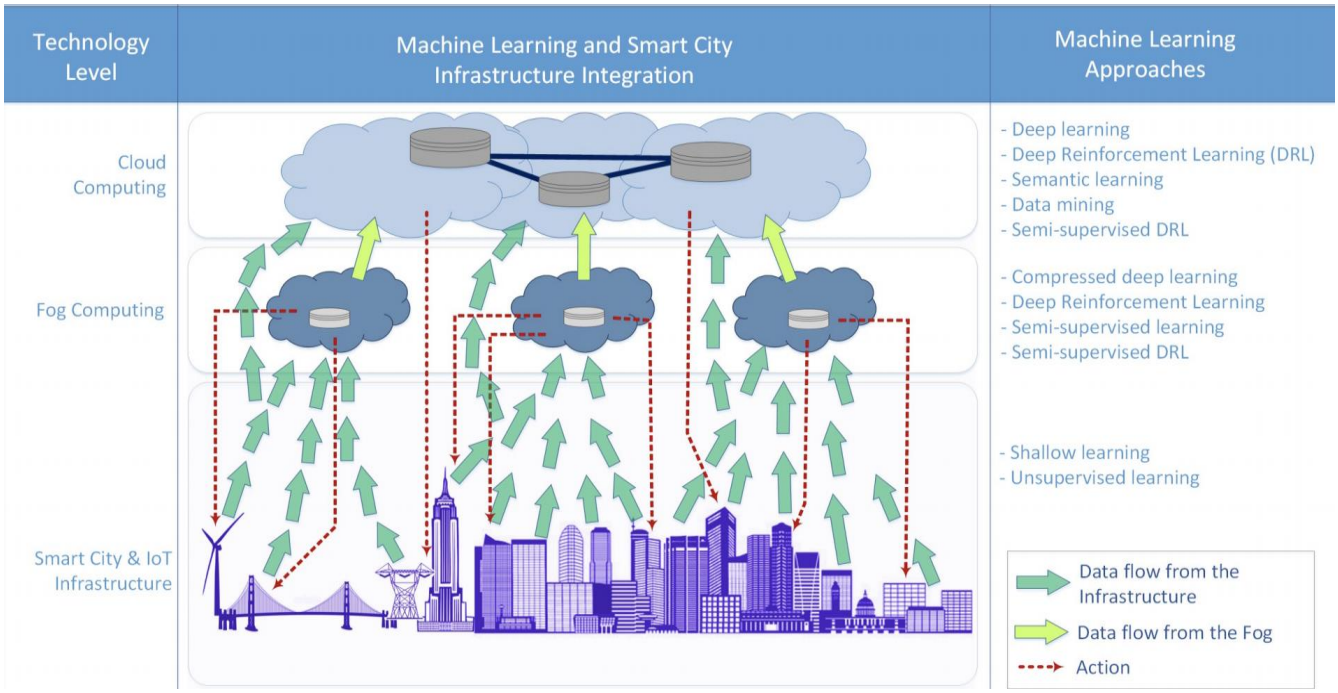


Рисунок 3. 1. Методи машинного навчання на різних рівнях системи розумного міста

Рівень IoT-інфраструктури – це рівень, який складається з датчиків та пристроїв з обмеженими ресурсами. Обмеження ресурсів не дозволяє розгортати комплексні моделі машинного навчання. Проте, на цьому рівні можна застосовувати деякі нескладні моделі, включаючи моделі навчання без учителя чи часткового навчання (наприклад, моделі, які використовують метод k-найближчих сусідів чи метод опорних векторів).

На рівні туманних обчислень сирі дані агрегуються та передаються на рівень хмарних обчислень. Стислі моделі глибинного навчання, DRL та методи часткового навчання можуть використовуватися, оскільки ресурси на цьому рівні мають менші обмеження порівняно з ресурсами IoT пристроїв.

На рівні хмарних обчислень більш складні та масштабні алгоритми машинного навчання та добування даних можуть бути інтегровані для отримання корисних знань та виявлення структурних закономірностей. Завдяки нещодавним досягненням в області розробки графічних процесорів (GPU), а також розробки ефективних алгоритмів ініціалізації параметрів нейронної

мережі, використання ReLU алгоритмів та впровадження довготривалої пам'яті (LSTM) нейронних мереж стало можливим вирішення проблеми зникання градієнту (vanishing gradient problem).

3.2 Компоненти хмарної системи

На рівні хмарної інфраструктури планується розгорнути сервіси, які займатимуться різними задачами:

1. IoT Gateway – сервіс, який займається прийомом повідомлень від об’єктів розумного міста, їх фільтрацією та маршрутизацією на відповідні сервіси. Являє собою вхідну точку для датчиків, сенсорів та інших пристроїв розумного міста. Побудований на базі платформи AWS IoT.
2. Сервіс Kafka. Слугує центральною шиною для відправки повідомлень між компонентами платформи. Складається з топіків повідомлень (наприлад, топіків для подій транспортної системи міста, системи вуличного освітлення, системи розумного паркування), які розміщені на вузлах кластера, що розгорнутий в хмарі.
3. Сервіс зберігання неструктурованих даних – сховище великої кількості даних на базі Hadoop кластера. Використовується для аналізу та виявлення закономірностей даних.
4. Сервіс аналітики та візуалізації даних. Надає інструменти для дослідження трендів даних розробленої системи. Використовується для довгострокового планування та розробки стратегії розвитку розумного міста.
5. Бази даних, які зберігають агреговані дані системи. Платформа розумного міста використовує різні бази даних залежно від задач, які потрібно вирішити.
6. Сервіси, які запускають моделі нейронних мереж та надають API для взаємодії.

7. Система відправки сповіщень адміністраторам платформи та кінцевим користувачам. Дозволяє відправляти повідомлення різними каналами: за допомогою інтегрованих мобільних додатків, SMS-повідомлень, електронної пошти.
8. API-сервіси, які надають засоби для розробки сторонніх клієнтських застосунків, інтеграції сторонніх розумних рішень.

На рис. 3. 2 зображено архітектуру компонентів системи, які розгорнуті в хмарному середовищі.

Рисунок 3. 2. Архітектура системи

3.3 Інструменти розробки

Основна мова програмування – Java 11. Це строго типізована мова, яка дозволяє створювати комплексні серверні програми використовуючи парадигму ООП. Додатки, розроблені на Java є кросплатформними та можуть запускатися на серверах з різними операційними системами.

При розробці серверної частини платформи застосовано різноманітні фреймворки та бібліотеки, які спрощують реалізацію системи. Spring Framework – фреймворк з відкритим кодом, який надає широкий спектр засобів створення промислових програмних систем:

1. IoC контейнер – модуль, який відповідає за конфігурацію та управління життєвим циклом об’єктів Java, ін’єкцію залежностей. Для налаштування особливостей роботи компонентів додатка можна використовувати XML файли або анотації Java.
2. Засоби аспектно-орієнтованого програмування, які дозволяють створювати наскрізні процедури (наприклад, компоненти для логування вхідних аргументів при виклику методів, вимірювання часу виконання методів компонентів, декорування додатковим функціоналом).
3. Доступ до даних. Бібліотеки Spring Data надають зручний API для роботи з різними реляційними та noSQL базами даних.
4. Декларативне управління транзакціями.
5. Модулі для розробки веб-додатків. Spring WebFlux – веб-фреймворк, який надає засоби для створення неблокуючих реактивних веб-додатків. Spring Web MVC – рішення, яке базується на використанні Servlet API та дозволяє швидко створювати RESTful сервіси.
6. Аутентифікація та авторизація. Завдяки інструментам, які надає проект Spring Security, можна легко налаштувати правила доступу до ресурсів системи.
7. Тестування програми. Для перевірки окремих компонентів програми та підсистем на коректність роботи, Spring Framework надає набір інструментів, за допомогою яких можна розробляти модульні та інтеграційні тести.

Для перетворення об’єктів використано бібліотеку Orika. Дана бібліотека надає зручний імперативний API для конфігурації особливостей перетворення об’єктів різних рівнів застосунку (наприклад, перетворення об’єктів model в об’єкти dto, які серіалізуються та віддаються користувачу REST API) та легко

інтегрується з Spring Framework, який представляє собою каркас серверного застосунку.

Завдяки використанню Lombok вдалось спростити написання Java Bean класів та уникнути шаблонного коду. Бібліотека Lombok – це плагін комплілятора Java, який дозволяє генерувати методи доступу до полів класу, методи `equals` та `hashcode`, конструктори, класи-будівники, аналізуючи Java анотації.

Для збірки проекту, автоматизації деяких задач при розробці системи (наприклад, розгортання тестового середовища для виконання інтеграційних тестів) використаний Gradle. Даний інструмент використовує Groovy / Kotlin DSL скрипти для конфігурації різних фаз життєвого циклу програми: компіляції, розгортання, тестування, аналізу коду тощо. Gradle дозволяє визначити різний набір залежностей для самої програми, тестових класів; є сумісним з артефактними репозиторіями Maven та Ivy.

На етапі проектування системи вирішено застосувати мікросервісний підхід. Таким чином, розроблена система складається з сукупності сервісів. Для гарантування постійної доступності системи потрібно реалізувати захисні механізми від поломки окремих компонентів. Resilience4j – бібліотека з відкритим кодом, яка надає різні шаблони захисту від несправностей у вигляді функцій вищого порядку (декораторів), які можна поєднувати між собою. У табл. 3.1 наведено використані компоненти бібліотеки.

Таблиця 3.1. Використані компоненти бібліотеки Resilience4j

Компонент	Принцип роботи	Опис
Retry	повторює операцію, яка виконалась неуспішно	багато несправностей є тимчасовими і можуть виправитись після короткої затримки
Circuit Breaker	тимчасово блокує можливі несправності	якщо система є перевантаженою та не може обробити клієнтські запити, кращим рішенням буде зупинити обслуговування клієнтів на певний час

Таблиця 3.1. Закінчення

Time Limiter	обмежує тривалість виконання операції	після певного часу очікування результату операції, ймовірність успішного виконання є досить малою, тому для раціонального використання ресурсів варто зупинити таку операцію
Cache	зберігає результат операції	деякі клієнтські запити до сервісів будуть ідентичними, тому для пришвидшення роботи сервісу варто зберігати результати складних операцій
Fallback	у випадку помилки виконання операції забезпечує альтернативний результат	у розподілених системах неможливо уникнути ситуацій несправностей, тому важливо розробити альтернативний спосіб обробки клієнтських запитів
Rate Limiter	обмежує кількість клієнтських запитів, які паралельно можуть оброблятися	для забезпечення SLA важливо обмежувати кількість запитів, які система може швидко обробити

Intellij IDEA – основне середовище написання коду сервісів системи, яке надає засоби редагування, рефакторингу та відладки програмних компонентів. IDE включає в себе інтеграції з багатьма популярними фреймворками та бібліотеками, базами даних; надає інструменти для запуску тестів, задач систем збірки Gradle та Maven. Існує підтримка синтаксису XML та YAML файлів.

3.4. Методи розробки програмного забезпечення

Розробка програмних модулів системи здійснювалась у відповідності до найкращих принципів та методів розробки програмного забезпечення.

Twelve-Factor App – методологія створення SaaS-застосунків, яка була використана на етапі кодування та тестування системи розумного міста. Основними особливостями систем, які розроблялись у відповідності до даної методології є:

- використання декларативного формату для автоматизації налаштування та розгортання;
- придатність до розгортання на сучасних хмарних платформах, що усуває необхідність у фізичних серверах та їх системному адмініструванні;
- мінімальна різниця між різними середовищами розгортання, що дозволяє застосувати принцип безперервного розгортання (continious delivery) та збільшує швидкість розробки;
- можливість горизонтально масштабуватись без необхідності зміни архітектури системи.

Весь код проекту зберігається в розподіленій системі контролю версій Git. При додаванні нового коміту CI система (continuous integration system) збирає проект, виконує модульні та інтеграційні тести, створює Docker image та зберігає його в приватній Docker registry системі.

Залежності системи повністю контролюються системою збірки Gradle. Визначено необхідні бібліотеки для компіляції проекту, розгортання в хмарному середовищі та виконання тестів.

Конфігурація сервісів зберігається у вигляді змінних оточення (environment variables). На відміну від конфігураційних файлів, які можна випадково зберегти в репозиторії коду, такий спосіб дозволяє легше змінювати конфігурацію сервісу між розгортаннями без модифікації коду.

Сервіси системи розумного міста запускаються як один або декілька процесів та не зберігають внутрішнього стану – являються stateless застосунками. Всі дані зберігаються в базах даних.

Для моніторингу за станом роботи сервісів використані інструменти логування. Всі повідомлення зберігаються та індексуються в сервісі Elasticsearch та доступні для аналізу.

3.5 Тестування системи

Протягом етапу розробки сервісів системи здійснювалось написання як модульних тестів, які перевіряють окремі компоненти на коректність, так і інтеграційних тестів, які тестують API сервісів та здійснюють перевірку основних бізнес-сценаріїв. Для написання модульних тестів використані бібліотеки JUnit 5, Mockito та модуль Spring Test. Розробка інтеграційних тестів відбувалась з використанням мови python та бібліотеки pytest. CI система запускала тести при додаванні нового коміту в репозиторій коду. Крім цього, здійснювалось ручне тестування функціональності системи наприкінці етапу розробки.

Вищезазначені методи тестування дозволили виявити дефекти системи на ранніх етапах розробки та впровадження системи та полегшити їх виправлення.

3.6 Модуль Керування автомобільним трафіком

На базі розробленої платформи розумного міста, яка вміє обробляти дані різних джерел, зберігати їх в одному форматі та інтегрувати для отримання максимальної користі, планується створення окремих модулів, які вирішують певні практичні задачі міста. Зокрема, в даній роботі приділено увагу аспектам проектування та розробки модуля Керування автомобільним трафіком. Основними джерелами даних модуля є датчики руху автомобілів та камери відеоспостереження встановлені на дорогах міста, а також дані місцеположення користувачів системи. На основі даних швидкості руху транспорту система визначає рівень заторів районів та окремих вулиць міста.

Відповідно до функціональних вимог система повинна зберігати дані користувача, його прокладені маршрути, історію пересувань, пошукових запитів та запланованих маршрутів. Для цих цілей використана нереляційна база даних MongoDB. Вона дозволяє зберігати дані у вигляді BSON документів, які поєднуються в колекції. На відміну від реляційних баз даних, записи не повинні

відповідати чіткій схемі, можуть містити довільні поля. Проаналізувавши предметну область системи, було визначено колекції документів, які потрібно зберігати в базі даних. Структуру деяких з них наведено нижче у табл. 3.2 - 3.5.

Таблиця 3.2. Структура документів колекції `users`

Назва поля	Опис	Тип даних
_id	унікальний ідентифікатор користувача	`objectId`
email	електронна адреса користувача; використовується для розсилки повідомлень на пошту (відновлення пароля)	`string`
password	пароль облікового запису користувача; зберігається в захешованому вигляді	`string`
first_name	ім’я користувача	`string`
last_name	прізвище користувача	`string`
language	мова інтерфейсу користувача	`string`
created_at	дата та час реєстрації користувача	`timestamp`
updated_at	дата та час модифікації інформації облікового запису користувача	`timestamp`
last_login_at	дата та час останнього логіну в систему	`timestamp`
last_location	координати останнього місцезнаходження користувача	`geoJSON object`
home_city_id	ідентифікатор міста, в якому мешкає користувач	`objectId`

Таблиця 3.3. Структура документів колекції `routes`

Назва поля	Опис	Тип даних
_id	унікальний ідентифікатор маршруту	`objectId`
user_id	ідентифікатор користувача, який створив даний маршрут; являється зовнішнім ключем	`objectId`
started_at	дата та час початку руху користувача по маршруту	`timestamp`
finished_at	дата та час завершення руху користувача по маршруту	`timestamp`
route_sections	список ділянок маршруту; елементи списку відображають частини маршруту	`array`
is_scheduled	поле, яке вказує чи являється даний маршрут запланованим	`bool`

Таблиця 3.4. Структура документів колекції `traffic_events`

Назва поля	Опис	Тип даних
_id	унікальний ідентифікатор події	`objectId`
city_id	ідентифікатор міста, у якому відбулась подія	`objectId`
district_id	ідентифікатор району, у якому відбулась подія	`objectId`
street_id	ідентифікатор вулиці, на якій відбулась подія	`objectId`
created_at	дата та час реєстрації події в системі	`timestamp`
resolved_at	дата та час вирішення (закінчення) події	`timestamp`
type	тип події; приймає значення: ROAD_ACCIDENT, CONGESTION, ROAD_REPAIR, OTHER	`string`
description	деталі події	`string`

Таблиця 3.5. Структура документів колекції `congestion_statistics`

Назва поля	Опис	Тип даних
_id	унікальний ідентифікатор документу, який представляє агреговану інформацію про рівень заторів певної території міста	`objectId`
district_id	ідентифікатор району міста, для якого зібрана статистика	`objectId`
datetime_range	інтервал часу, за який зібрана статистика	`object`
congestion_level	рівень заторів у районі, %	`int`
traffic_jams_count	кількість заторів у районі	`int`
traffic_jams_length	довжина заторів на вулицях, км	`int`
extra_travel_time	середній додатковий час, який потрібно витрати на поїздку, хв	`int`

Для ефективної роботи з базою даних використана бібліотека Spring Data MongoDB. Вона являє собою ORM рішення та надає інструменти як для базових операцій, так і для створення складних запитів до бази даних. Оскільки MongoDB не забезпечує цілісності зв’язків між записами подібно до того, як це роблять реляційні бази даних, відповідна логіка реалізована на рівні серверного додатку.

Для прикладу, компонент, який взаємодіє з базою даних та містить операції над документами колекції `routes`, являє собою інтерфейс, який розширює базовий інтерфейс `CrudRepository<T, ID>`, що надає базові можливості для взаємодії з базою даних:

```
@Repository
public interface RouteRepository
    extends MongoRepository<Route, ObjectId> {

    List<Route> findByUserOrderByCreatedAt(User user);

    List<Route> findByUserAndIsScheduled(User user, Boolean scheduled);

    int countByUser(User user);
}
```

При запуску серверного додатку бібліотека Spring Data MongoDB сконструює клас, який імплементує створений інтерфейс та помістить об’єкт класу в IoC контейнер. Таким чином, компоненти рівня бізнес-логіки зможуть його використовувати для реалізації операцій з маршрутами користувачів: створення нового маршруту, відображення деталей маршруту тощо.

3.7 API для розробників сторонніх додатків

В рамках роботи розроблений прикладний програмний інтерфейс для доступу до функціональності системи. Сторонні додатки можуть взаємодіяти з системою через REST API, який базується на можливостях HTTP протоколу та накладає певні обмеження на архітектуру системи, зокрема:

1. Клієнт-серверна архітектура. Відокремлюючи проблеми репрезентації даних, інтерфейсу користувача, від проблем зберігання та обробки даних, API серверних застосунків стає більш універсальним для різних клієнтських застосунків (веб додатків, мобільних застосунків).
2. Відсутність стану на сервері. Будь-який запит клієнта до сервера повинен містити всю необхідну інформацію для обробки запиту. Дані, пов’язані з сесією користувача, зберігаються на клієнті.
3. Кешування. Сервер повинен зазначати, яку інформацію клієнт може зберегти в себе локально та використати пізніше.
4. Універсальний інтерфейс. Архітектура системи повинна задовольняти наступні обмеження: ресурси повинні ідентифікуватись, модифікація

стану ресурсів відбувається через їх репрезентації, повідомлення системи є інформативними.

5. N-рівнева система. Серверний застосунок повинен складатись з ієрархії сервісів, кожен з яких відповідає за вирішення визначених задач (робота з базою даних, бізнес-логіка, інтеграція з сторонніми сервісами).

Результатом роботи є API для управління обліковим записом користувача, доступу до інформації про рівень заторів, надзвичайні ситуації певної території міста, побудови маршруту, перегляду історії створених маршрутів, детальної інформації вибраного маршруту, створення поїздки у вибраний час. У табл. 3.6 наведено приклади деяких доступних можливостей системи.

Таблиця 3.6. API модуля Керування автомобільним трафіком

URI	HTTP метод	Опис
api/users	POST	Створити обліковий запис користувача
api/users/{id}	GET	Переглянути деталі облікового запису користувача з унікальним ідентифікатором id
api/users/{id}	PUT	Оновити інформацію облікового запису
api/login	POST	Авторизувати користувача за допомогою електронної пошти і пароля; система повертає токен авторизації, з яким клієнт повинен здійснювати наступні запити до сервера
api/cities/{cityId}/districts/{districtId}/congestion-level	GET	Отримати інформацію про рівень заторів у вибраному районі міста; результат містить дані про середню швидкість транспорту, найбільш завантажені вулиці
api/cities/{cityId}/districts/{districtId}/events	GET	Отримати інформацію про події у вибраному районі міста: аварії, ремонти доріг, розміщення поліцейських патрулів
api/routes	GET	Отримати список оптимальних маршрутів за параметрами: координатами чи адресою місця перебування та місця призначення, видом транспорту тощо
api/users/{userId}/routes/{routeId}	PUT	Асоціювати вибраний маршрут з користувачем

Таблиця 3.6. Закінчення

api/users/{userId}/routes	GET	Переглянути маршрути користувача; параметри запиту можуть вказувати період маршрутів, місто, в якому система створила маршрут тощо
api/users/{userId} /scheduled-routes	POST	Створити запланований маршрут. вказуються деталі маршруту (початок та кінець маршруту, дата та час початку маршруту)

Висновки по розділу

В даному розділі описаний процес розробки системи розумного міста. Наведена архітектура платформи, основні компоненти системи, їх особливості. Викладені аспекти реалізації модуля Керування автомобільним трафіком. Звернено увагу на основні сутності системи. Для розробників сторонніх прикладних застосунків реалізовано REST API доступу до функціоналу системи.

РОЗДІЛ 4. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ

4.1 Опис ідеї проекту

Таблиця 4.1. Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користучача
Пропонується система розумного міста, яка використовуючи технології Інтернету Речей, алгоритми машинного навчання та прогнозування автоматизує процеси міста, управляє обмеженими ресурсами та надає інструменти аналізу поточного стану міста.	Управління автомобільним трафіком, знаходження оптимального маршруту у межах міста	Впровадження системи дозволить владі міста ефективніше використовувати та розвивати інфраструктуру міста (дороги, мережу світлофорів). Жителі міст зможуть швидше пересуватись по місту. Компанії зможуть заощадити кошти на перевезення товарів.
	Моніторинг екологічної ситуації в місті	

Таблиця 4.2. Опис ідеї стартап-проекту

№	Техніко-економічні характеристики ідеї	Продукція конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	IBM Watson	Google Асистент	HPE Heaven			
1	Економічність					Великі витрати на встановлення інфраструктури на базі якої буде працювати система (датчики, камери відеоспостереження)	Система розгортатиметься в хмарному середовищі, а отже витрати на підтримку середовища розгортання є мінімальними	Зменшення витрат в галузях, процеси яких автоматизуються системою (напр., перевезення вантажів та людей)

Таблиця 4.2. Закінчення

2	Безпека					Можливі атаки на систему та викрадення персональних даних користувачів	Застосування стандартних засобів захисту інформації хмарних провайдерів	Реалізація додаткових методів захисту чутливої інформації від зловмисників
3	Надійність					Використання сервісів, обчислювальних ресурсів сторонніх компаній	Використання інструментів для моніторингу стану компонентів системи у режимі реального часу	Імплементація механізмів захисту та стійкості до відмов окремих компонентів для безперервної роботи системи в цілому
4	Технологічність					Необхідність підтримки датчиків та камер різних виробників	Застосування стандартних засобів побудови програмних застосунків	Можливість стороннім розробникам розширяти функціональність системи підключаючи свої рішення

4.2 Технологічний аудит ідеї проекту

Таблиця 4.3. Технологічна здійсненність ідеї проекту

№	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Створення платформи розумного міста для інтеграції даних пристроїв міста	Java та IoT фреймворк Eclipse Leshan	Наявна	Доступня та безкоштовна
2		.NET Core та Universal Windows Platform	Наявна. Може використовуватись на обмеженому ряді пристроїв	Доступня та безкоштовна

Таблиця 4.3. Закінчення

3		Python та Google Cloud IoT Platform	Наявна	Доступня; IoT платформа є платною послугою Google Cloud
Обрана технологія реалізації ідеї проекту:1				

Висновок: технологічна реалізація продукту – можлива, вибрана технологія №1.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Таблиця 4.4. Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн./ум.од	100 млн. грн./рік
3	Динаміка ринку	Стрімко зростає
4	Наявність обмежень для входу	Необхідність сучасної інфраструктури в містах, де інтегрується система
5	Специфічні вимоги до стандартизації та сертифікації	Проходження сертифікації надійності зберігання приватних даних користувачів
6	Середня норма рентабельності в галузі або по ринку, %	20%

Висновок: враховуючи кількість головних гравців по ринку, зростаючу динаміку ринку, невелику кількість конкурентів та середню норму рентабельності можна зробити висновок, що на даний момент, ринок для входження стартап-продукту є привабливим.

Таблиця 4.5. Характеристика потенційних клієнтів стартап-проекту

№	Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп клієнтів	Вимоги споживачів до товару
1	Вирішення проблеми заторів та насиченого автомобільного трафіку у містах	Міська влада, водії автомобілів, жителі міст	Міська влада прагне покращити інфраструктуру міста; Водії автомобілів зменшити час поїздок та витрати на паливо; Жителі міст зменшити час поїздок.	До системи: стабільна робота, ефективне визначення оптимальних маршрутів, сповіщення. До компанії: оновлення та покращення функціональності.

Таблиця 4.6. Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренти	Наявність конкурентів котрі надають схожі рішення	Зменшення ціни на сервіси системи; Розробка унікальних функцій системи; Надання ліцензій на обслуговування.
2	Кошти на розробку та підтримку продукту	Закінчення грошей та недостатнє фінансування	Залучення додаткових інвесторів, мотивація роботи на перспективу; Ітеративна розробка продукту задля покрокового виведення продукту на ринок та отримання відповіді користувачів. Залучення коштів у міської влади.
3	Вихід аналогу	Вихід аналогу даного товару може призвести до переключення уваги на нову систему та її поширення	Вихід товару на ринок в коротші строки з не повною, але достатньою, функціональністю для зацікавлення усіх цільових аудиторій; Проведення рекламної компанії

Таблиця 4.7. Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Новий продукт	Вихід на ринок, Зменшення монополії, Надання нових рішень у сфері управління інфраструктурою міст	Розробка нової функціональності; Вихід нової продукції на ринок; Надання різноманітних типів ліцензій в залежності від потреб замовників.
2	Вихід аналогу	Надати продукт з певними характеристиками та можливостями що відсутні у компаній конкурентів	Аналіз ринку та користувачів задля задоволення їх потреб та надання функціональності у найкоротші строки за ціну, котра є дешевшою ніж у продуктів-замінників.
3	Зворотній зв’язок від користувачів	Можливість отримання необхідної інформації для вдосконалення продукту	Наявність вхідних даних та реакція на них з боку команди розробників задля задоволення потреб та бажань кінцевих користувачів системи.
4	Грошова винагорода за рекламу	При достатньому попиту на систему можлива комерціалізація продукту на основі реклами задля отримання грошової винагороди для подальшого розвиту продукту та оплати заробітної плати працівникам	Точкова комерціалізація продукту; Введення реклами; Ведення додаткових коштів у проект задля його подальшого розвитку.

Таблиця 4.8. Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	Тип конкуренції: монополістична	Товар від кожної компанії на ринку, являється недосконалим замінником товару, реалізованого іншими фірмами; На ринку є умови для входу та виходу; Ціна корелює між суперниками;	Розробка продукту з характеристиками, які покривають сфери вживання що не покривають інші товари-замінники; Кореляція цін у відповідності до товарів замінників; Різни типи ліцензій.

Таблиця 4.8. Закінчення

2	Рівень конкурентної боротьби: світовий	Всі продукти замінники розроблялись інтернаціональними командами з різних куточків світу, продукти не належать до певної держави, а належать команді розробників	Вихід на ринок збуту продукту з клієнто-необхідною функціональністю; Налагодження маркетингу на основних Інтернет ресурсах задля охоплення великої кількості потенційних користувачів; Надання бета-версій продукту.
3	Галузева ознака: внутрішньогалузева	Даний тип продукту може використовуватися тільки у сфері розробки IT додатків \ продуктів	Надання зручного, інтуїтивно зрозумілого інтерфейсу; Підтримка всім відомих методів взаємодії з середовищем розробки; Наявність документації та он-лайн підтримки.
4	Конкуренція за видами товарів: товарно-видова	Дана конкуренція – конкуренція між товарами одного виду.	Впровадження функціональності яка відсутня у товарів-замінників; Спрощення інтерфейсів; Надання підтримки.
5	Характер конкурентних переваг: цінова та не цінова	Цінові переваги – точкова комерціалізація; Не цінова – надання функціональності, що відсутня у товарах-замінниках.	Надання платних ліцензій лише на критично важливу функціональність для клієнта з певним строком підтримки, що зазначена у відповідній ліцензії; Впровадження унікальної функціональності.
6	За інтенсивністю: марочна	Наявність унікального знаку що відрізняє даний продукт від продуктів-замінників	Впровадження власної назви та власного знаку.

Таблиця 4.9. Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	Компанії, які надають сервіс онлайн карт з можливостями побудови маршрутів (Google Maps, maps.me)	Бар'єром входження на ринок є наявність базового функціоналу	Цінова політика та якість програмних рішень	Потреба у стабільному програмному забезпеченні за низькою ціною	Конкуренти можуть розпочати розробку нового програмного забезпечення

Таблиця 4.9. Закінчення

Висновки	Інтенсивність конкурентної боротьби середня	Є можливість швидкого виходу на ринок; Потенційних конкурентів немає; Строк виходу на ринок – 2 роки.	Постачальники диктують умови стосовно оплати ліцензій	Клієнти диктують умови роботи на ринку шляхом встановлення вимог до ціни та функціональної частини систем	Можливі обмеження в роботі продукту на ринку через впровадження інших систем зі сторони держави

Проаналізувавши можливості роботи на ринку з огляду на конкурентну ситуацію можна зробити висновок: оскільки кожний з існуючих продуктів не впливає у великій мірі на поточну ситуацію на ринку в цілому, кожний з існуючих продуктів має свою специфічну сферу використання та свої позитивні та негативні сторони щодо рішення певних типів задач, то робота та вихід на даний ринок є можливою і реалізованою задачею.

Для виходу на ринок продукт повинен мати функціонал що відсутній у продуктів-аналогів, повинен задовольняти потреби користувачів, мати необхідний та достатній функціонал з конфігурування, підтримку зі сторони розробників та можливість розробки спеціального функціоналу за відповідною ліцензією.

Таблиця 4.10. Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Економічний	Забезпечення користувачів безкоштовним базовим функціоналом з можливістю розширення за рахунок купівлі підписок на окремі компоненти системи
2	Технологічний	Використання сучасних технологій дозволить розвивати систему протягом довгого часу з можливістю швидкого розширення та стійкістю до змін

Таблиця 4.10. Закінчення

3	Політичний	Співпраця з країнами-союзниками дозволить розширити область застосування програмного рішення
12	Правовий	Підтримка інноваційної діяльності зі сторони міської влади
5	Соціально-психологічний	Звільнення працівників від буденної праці та надання часу для виконання творчих завдань
6	Організаційно-управлінський	Надання сучасних технологій керівникам для максимальної оптимізації ефективності роботи працівників
7	Безпека	Моніторинг стану доріг та зменшення кількості дорожньо-транспортних аварій

Таблиця 4.11. Порівняльний аналіз сильних та слабких сторін системи

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з запропонованим						
			-3	-2	-1	0	+1	+2	+3
1	Економічний	15					+		
2	Технологічний	19			+				
3	Політичний	13				+			
4	Правовий	18				+			
5	Соціально-психологічний	17					+		
6	Організаційно-управлінський	16		+					
7	Безпека	20		+					

Таблиця 4.12. SWOT аналіз стартап-проекту

Сильні сторони (S): – Різноманітні рекламні акції за рахунок вкладників та інвесторів. – Застосування новітніх технологій під час розробки. – Ефективна кадрова політика, професійність та кваліфікованість кадрів. – Постійне оновлення продукції, дослідження, направлені на покращення якості продукції.	Слабкі сторони (W): – Недостатній імідж підприємства на ринку. – Необхідність придбання великого об'єму прав на користування продукцією. – Виникнення додаткових витрат на навчання та перекваліфікацію персоналу. – Можливі додаткові витрати на навчання персоналу під час розробки.
Можливості (O): – Зростання попиту на продукцію. – Вихід на нові ринки або сегменти. – Послаблення позицій конкурентів. – Використання новітніх світових технологій. – Залучення висококваліфікованого персоналу. – Послуги даного типу надзвичайно популярні у наш час.	Загрози (T): – Зміна політики непрямих конкурентів. – Нестабільна політична та економічна ситуація. – Наявність прямих конкурентів. – Зниження доходів потенційних споживачів. – Не прийняття новизни проекту та прихильність до інших брендів.

Таблиця 4.13. Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Безкоштовне надання певного функціоналу у користування споживачам на обмежений термін	Головний ресурс – люди, даний ресурс - наявний	2-3 місяці
2	Реклама	Залучення власних коштів для реклами товару	1-2 місяці
3	Написання статей та опис товару на відомих ресурсах	Головний ресурс – час, даний ресурс - наявний	2-3 тижні
4	Презентація товару на хакатонах й інших ІТ заходах	Ресурс – час та гроші для участі, наявні	1-3 місяці

4.4 Розроблення ринкової стратегії проекту

Таблиця 4.14. Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Влада метрополісів, які характеризуються загостренням автомобільної ситуації	Користувачі мають високий ступінь готовності, оскільки вже використовують певні технологічні засоби для вирішення проблеми	Високий рівень попиту у середніх та великих містах з можливим розширенням на регіони	Конкуренція середня	Рівень простоти входу в сегмент є середнім
2	Адміністрація міст, які бажають ефективніше використовувати ресурси	Високий ступінь готовності	Очікується високий рівень попиту	Конкуренція не інтенсивна у країні, але має середній рівень інтенсивності у Європі	Рівень простоти входу в сегмент є середнім
Які цільові групи обрано:1, 2					

Відповідно до проведеного аналізу можна зробити висновок, що підходящою цільовою групою для розповсюдження даного програмного продукту є влада міст, які намагаються вирішити проблему інтенсивного автомобільного трафіку в місті або покращити ефективність використання ресурсів міста. Відповідно до стратегії охоплення ринку збуту товару обрано стратегію масового маркетингу, оскільки адміністрації міст надається стандартизований продукт з можливістю розширення функціональності за домовленістю (відповідно до ліцензії).

Таблиця 4.15. Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання функціональності, що відсутня у товарів-замінників, підтримка клієнтів	Проведення реклами, освітлення унікальної функціональності через інтернет ресурси та інші канали, контакт напряму з споживачами; формування лояльності і прихильності споживачів	Зниження ступеню замінності товару; Прихильність клієнтів; Відмітні властивості товару; Відмітні характеристики товару;	Стратегія диференціації

Таблиця 4.16. Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки
Ні, оскільки є товари-замінники, але дані товари замінники не мають деякого необхідного функціоналу	Так, ціль компанії знайти нових споживачів та, частково, забрати існуючих у конкурентів задля задоволення потреб останніх	Компанія частково копіює характеристики товару конкурента, основна ціль компанії розробка нового унікального функціоналу, з підтримкою основного функціоналу конкурентів	Стратегія заняття конкурентної ніші

Таблиця 4.17. Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Вирішення проблеми інтенсивного трафіку в місті	Стратегія спеціалізації	Надання водіям програмного застосунку для прокладання оптимальних маршрутів	Моніторинг стану транспортної системи, кількості аварій у районах міста
2	Лідерство по витратах, а саме безкоштовна використання у поєднанні з підписками	Стратегія лідерства по витратах	Забезпечення безкоштовного базового функціоналу з подальшим розширення за рахунок купівлі підписок на окремі сервіси	Безкоштовний функціонал, окремі підписки на необхідні сервіси

Відповідно до проведеного аналізу можна зробити висновок, що стартап-компанія вибирає як базову стратегію розвитку – стратегію диференціації, як базову стратегію конкурентної поведінки – стратегію заняття конкурентної ніші.

4.5 Розроблення маркетингової програми стартап-проекту

Таблиця 4.18. Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Знаходження оптимального маршруту в умовах міста	Програмні застосунки для водіїв та жителів міста з можливістю задати маршрут та отримати результат	Можливість інтерактивного та зручного керування даними з інтеграцією з іншими частинами системи
2	Моніторинг стану транспортної системи	Веб-сервіс з картою стану заторів, середньою швидкістю руху транспорту	Висока точність даних; наявність інструментів візуалізації та аналітики даних

Таблиця 4.18. Закінчення

3	Керування паркувальними місцями	Сервіс бронювання паркувальних місць з можливістю оплати для водіїв	Відображення місцезнаходження парковок, кількості вільних паркувальних місць
4	Моніторинг екологічної ситуації	Система сповіщення жителів міста	Можливість налаштування критеріїв надсилань сповіщень жителям міста

Таблиця 4.19. Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
1. Товар за задумом	Система розумного міста на базі нейронних мереж		
2. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх/Тл/Е/Ор
	Вартість обслуговування	Є низькою за рахунок використання провайдерів хмарної інфраструктури	Вигідно для клієнтів системи
	Здатність відновлюватися після настання відмови компонентів системи	Система спроектована для боротьби з непередбаченими відмовами	Покращує вибір потенційних клієнтів
	Надійне зберігання приватної інформації користувачів	Застосовуються передові технології захисту інформації	Є обов’язковою вимогою замовників
3. Товар із підкріпленням	До продажу: наявна повна документація, акції на придбання декількох ліцензій, знижки для певних сегментів на покупку товару		
	Після продажу: додаткова підтримка спеціалістів налаштування, підтримка з боку розробника		
За рахунок чого потенційний товар буде захищено від копіювання: захист інтелектуальної власності, патент			

Таблиця 4.20. Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
15-25 тис. грн/міс	12-22 тис. грн/міс	40-50 тис. грн/міс	10-30 тис. грн/міс

Таблиця 4.21. Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Влада міст закуповуватиме ліцензію на систему через державні тендери	Встановлювати систему на сервери адміністрації міста чи допомагати розгортати в хмарному середовищі	Оптимальна	Власними силами

Таблиця 4.22. Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Обирають системи, які принесуть користь місту	Інтернет, газети та публіцистичні журнали	Ефективність роботи	Викликати бажання клієнта придбати товар	Автоматизація процесів та зменшення витрат на підтримку інфраструктури
2	Працюють через системи держзакупівель		Значний вплив	Донести важливу інформацію про проблеми, які вирішує система	

Як результат було створено ринкову (маркетингову) програму, що включає в себе визначення ключових переваг концепції потенційного товару, опис моделі товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

Висновки по розділу

В четвертому розділі описано стратегії та підходи з розроблення стартап-проекту, визначено наявність попиту, динаміку та рентабельність роботи ринку, як висновок було вказано що існує можливість ринкової комерціалізації проекту. Розглянувши потенційні групи клієнтів, бар’єри входження, стан конкуренції та конкурентоспроможність проекту було встановлено що проект є перспективним. Розглянуто та вибрано альтернативу впровадження стартап-проекту та доведено доцільність подальшої імплементації проекту.

ВИСНОВКИ

У ході виконання магістерської дисертації досліджено концепцію розумного міста. За результатами аналізу проблем сучасних міст та існуючих рішень сформульовано завдання роботи, обґрунтовано актуальність розроблюваної системи розумного міста. Описані вимоги та проблеми, які потрібно розв’язати при проектуванні та розробці системи.

На етапі проектування платформи проведений аналіз доступних інформаційних технологій, розглянуті їх переваги та обмеження. Досліджено доступні технології Інтернету Речей. Вивчено наявні методи машинного навчання, та, в результаті, вибрано моделі, які найкраще підходять для вирішення задач розумного міста. Для потокової обробки даних вибрано рішення Apache Kafka, для зберігання неструктурованих даних – Apache Hadoop.

Описана архітектура системи, яка складається з рівня IoT інфраструктури, рівня туманних обчислень та рівня хмарних обчислень. Наведені основні компоненти, які складають хмарну платформу.

На основі створеної платформи розумного міста розроблено модуль Керування автомобільним трафіком. Головними можливостями модуля є пошук оптимального маршруту в межах міста, відображення поточного стану автомобільного трафіку в місті, рівня заторів та середньої швидкості руху транспорту на окремих ділянках вулиць. Для надання стороннім розробникам можливості створювати прикладні програмні застосунки розроблено прикладний програмний інтерфейс у вигляді REST API.

Проведений маркетинговий аналіз стартап-проекту. Проведено опис ідеї проекту, технологічний аудит ідеї проекту, аналіз ринкових можливостей запуску стартап-проекту. Розроблено ринкову стратегію проекту та маркетингову програму стартап-проекту.

Результати роботи над магістерською дисертацією опубліковані у науковій статті.

Наукова новизна одержаних результатів магістерської дисертації полягає у наступному:

вперше:

- здійснено аналіз та комплексне порівняння наявних комерційних рішень та наукових розробок у сфері систем розумного міста;
- застосовано DRL модель часткового навчання для вирішення задач розумного міста, зокрема для пошуку оптимального маршруту в умовах міста;
- розгорнуто нескладні нейронні мережі на інфраструктурі туманних обчислень.

удосконалено:

- покращено ефективність наявних методів пошуку оптимального маршруту за рахунок використання машинного навчання та методів прогнозування;
- покращено інтеграцію різних джерел даних в межах платформи завдяки використанню платформи Apache Kafka;
- удосконалено інтеграцію розумних IoT пристроїв з серверними застосунками.

здобуло подальший розвиток:

- обробка на аналіз неструктурованих даних системи;
- розробка системи сповіщення користувачів;
- реалізація публічного API доступу до функціоналу системи;
- реалізація механізмів захисту персональних даних користувачів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Імас О., Бугар А. Теорія рішень «розумного» міста та можливості її реалізації на базі єдиної муніципальної платформи, 2019. – Режим доступу: https://hub.kyivstar.ua/ua/teoriya-resheniy-umnogo-goroda-i-vozmozhnosti-ee-realizatsii-na-baze-edinoy-munitsipalnoy-platformyi/.
2. From transport to street lighting: The emergence of smart city platforms. – Retrieved from: https://www.euractiv.com/section/digital/opinion/tue-from-transport-to-street-lighting-the-emergence-of-smart-city-platforms/.
3. Harrison C., Eckman B., Hamilton R. Foundations for Smarter Cities. IBM Journal of Research and Development, 2010, Vol. 54, P. 1-16.
4. Smart City Platforms: The Intelligent Core of Smart Cities, Beecham Research, 2015. – Retrieved from: http://beechamresearch.com/download.aspx?id=44.
5. A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, and M. Mohammadi, «Toward better horizontal integration among IoT services», IEEE Communications Magazine, vol. 53, no. 9, pp. 72–79, 2015.
6. P. Vlacheas, R. Giaffreda, V. Stavroulaki, D. Kelaidonis, V. Foteinos, G. Poulios, P. Demestichas, A. Somov, A. R. Biswas, and K. Moessner, «Enabling smart cities through a cognitive management framework for the internet of things», IEEE Communications Magazine, vol. 51, no. 6, pp. 102–111, 2013.
7. Q. Wu, G. Ding, Y. Xu, S. Feng, Z. Du, J. Wang, and K. Long, «Cognitive Internet of Things: a new paradigm beyond connection», IEEE Internet of Things Journal, vol. 1, no. 2, pp. 129–143, 2014.
8. S. Feng, P. Setoodeh, and S. Haykin, «Smart Home: Cognitive Interactive People-Centric Internet of Things», IEEE Communications Magazine, vol. 55, no. 2, pp. 34–39, 2017.
9. B. Tang, Z. Chen, G. Hefferman, S. Pei, W. Tao, H. He, and Q. Yang, «Incorporating Intelligence in Fog Computing for Big Data Analysis in Smart Cities», IEEE Transactions on Industrial Informatics, vol. PP, no. 99, pp. 1–11, 2017.

10. Greengard S. The Internet of Things, MIT Press, 2015, 230 p.
11. Kapoor A. Hands-On Artificial Intelligence for IoT: Expert machine learning and deep learning techniques for developing smarter IoT systems, Packt Publishing, 2019, 390 p.
12. Kranz M. Building the Internet of Things: Implement New Business Models, Disrupt Competitors, Transform Your Industry, Wiley, 1 edition, 2016, 272p.
13. Slama D., Puhlmann F., Morrish J., Bhatnagar R. M. Enterprise IoT: Strategies and Best Practices for Connected Products and Services, O'Reilly Media; 1 edition, 2015, 492 p.
14. Stackowiak R. Big Data and The Internet of Things: Enterprise Information Architecture for A New Age, Apress; 1st ed. edition, 2015, 220 p.
15. Washburn D., Sindhu U., Balaouras S. Helping CIOs Understand «Smart City» Initiatives: Defining the Smart City, Its Drivers, and the Role of the CIO. Cambridge, MA, Forrester Research Inc., 2010.
16. Woods E. Smart City Platforms: IoT, Digital Solutions, and Data Technologies Enabling the City as a Service. Navigant Research, 2018. – Retrieved from: https://www.navigantresearch.com/reports/smart-city-platforms.
17. Головко В.А. Нейронные сети: обучения, организация и применение. М.: ИПРЖР, 2008.
18. Каллан. Р. Основные концепции нейронных сетей. Изд-во: Вильямс, 2002, 287 с.
19. Крисилов В.А. Представление исходных данных в задачах нейросетевого программирования. Одесса: ОНПУ, 2003.

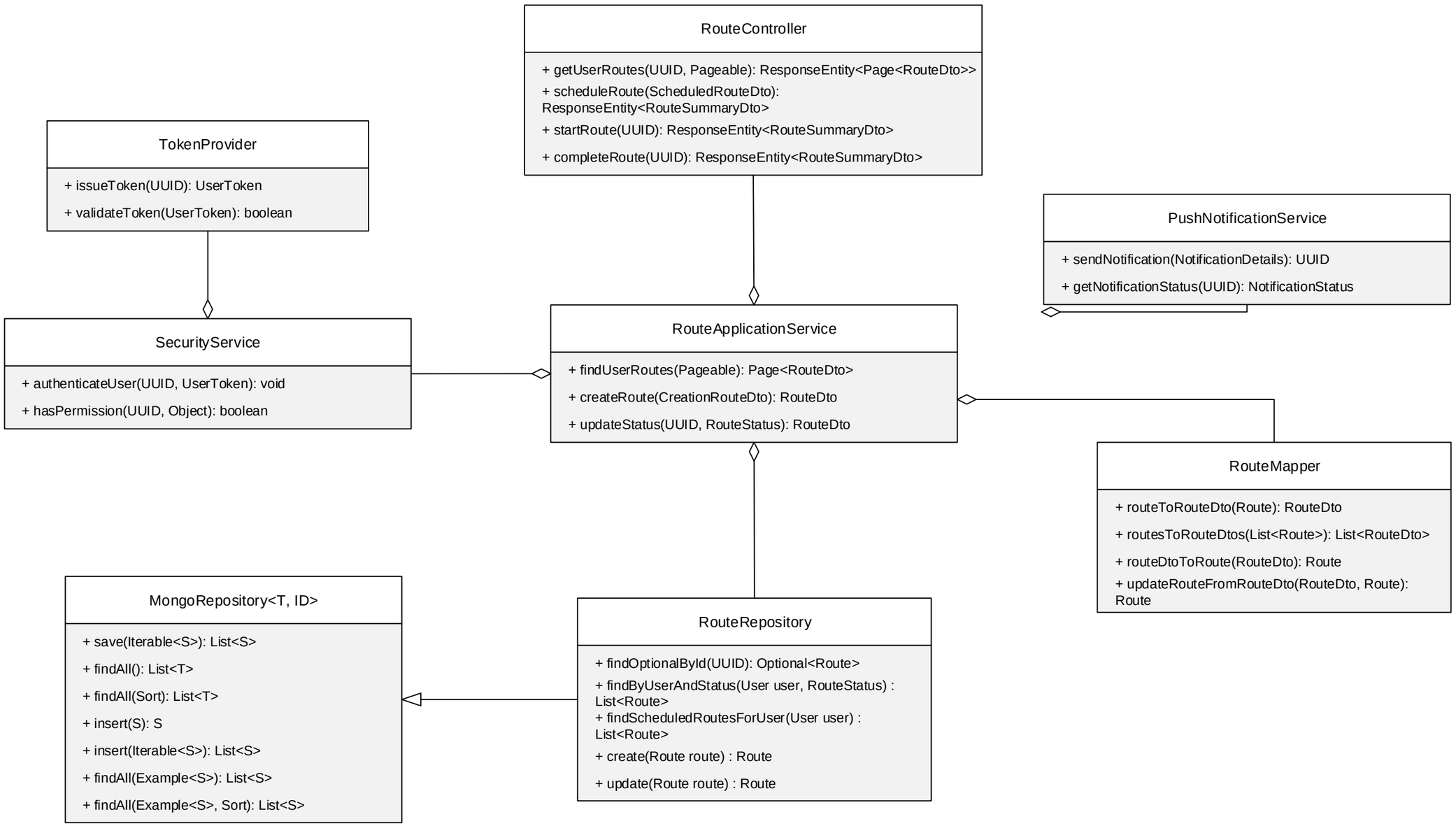
ДОДАТОК А

Результати перевірки на співпадіння

ДОДАТОК Б

Плакати та графічні матеріали

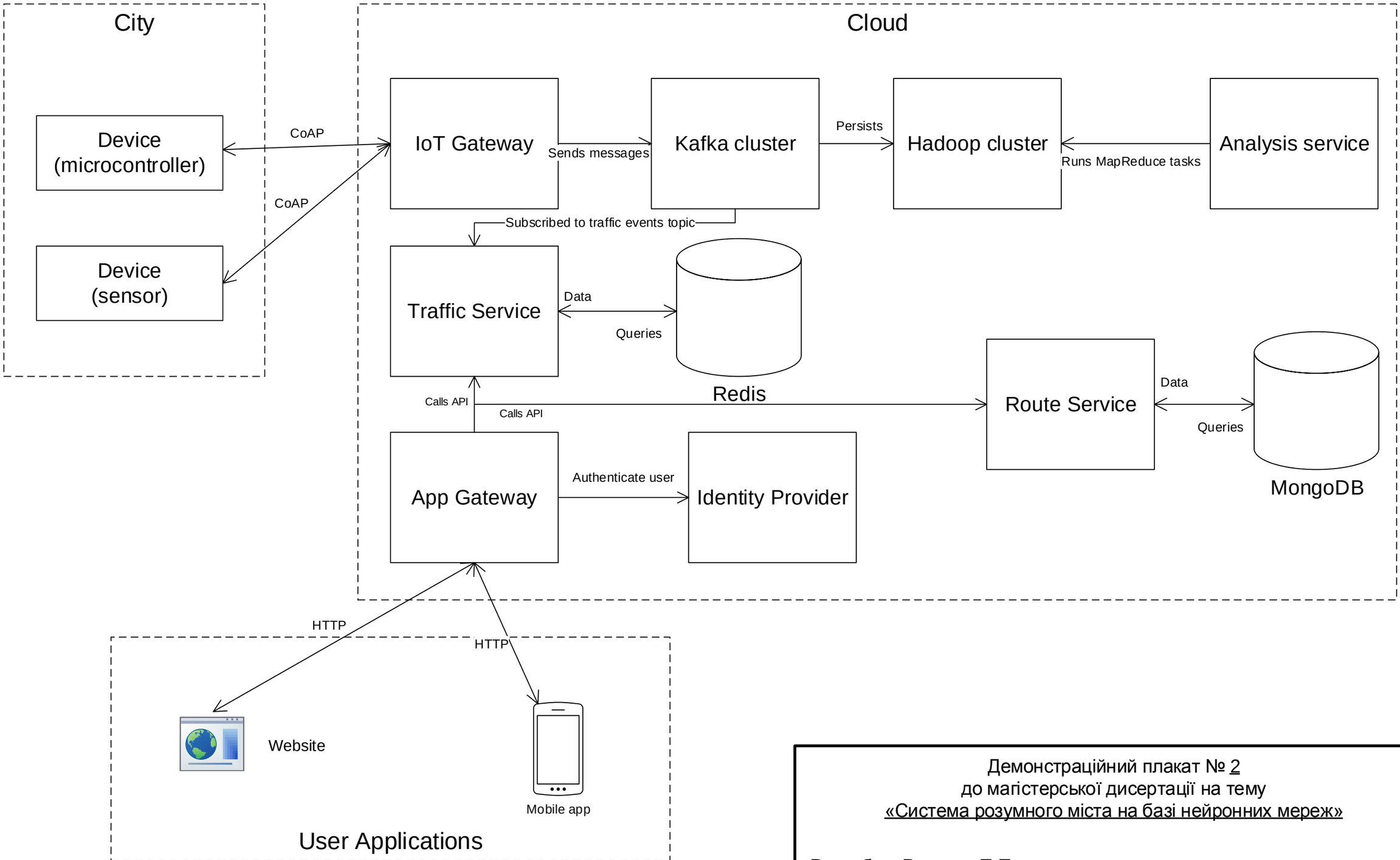
Діаграма класів модуля управління маршрутами



Демонстраційний плакат № 1
до дипломної роботи на тему
«Система розумного міста на базі нейронних мереж»

Розробив: Роспопа П.П.
Прийняв: Сирота О.П.

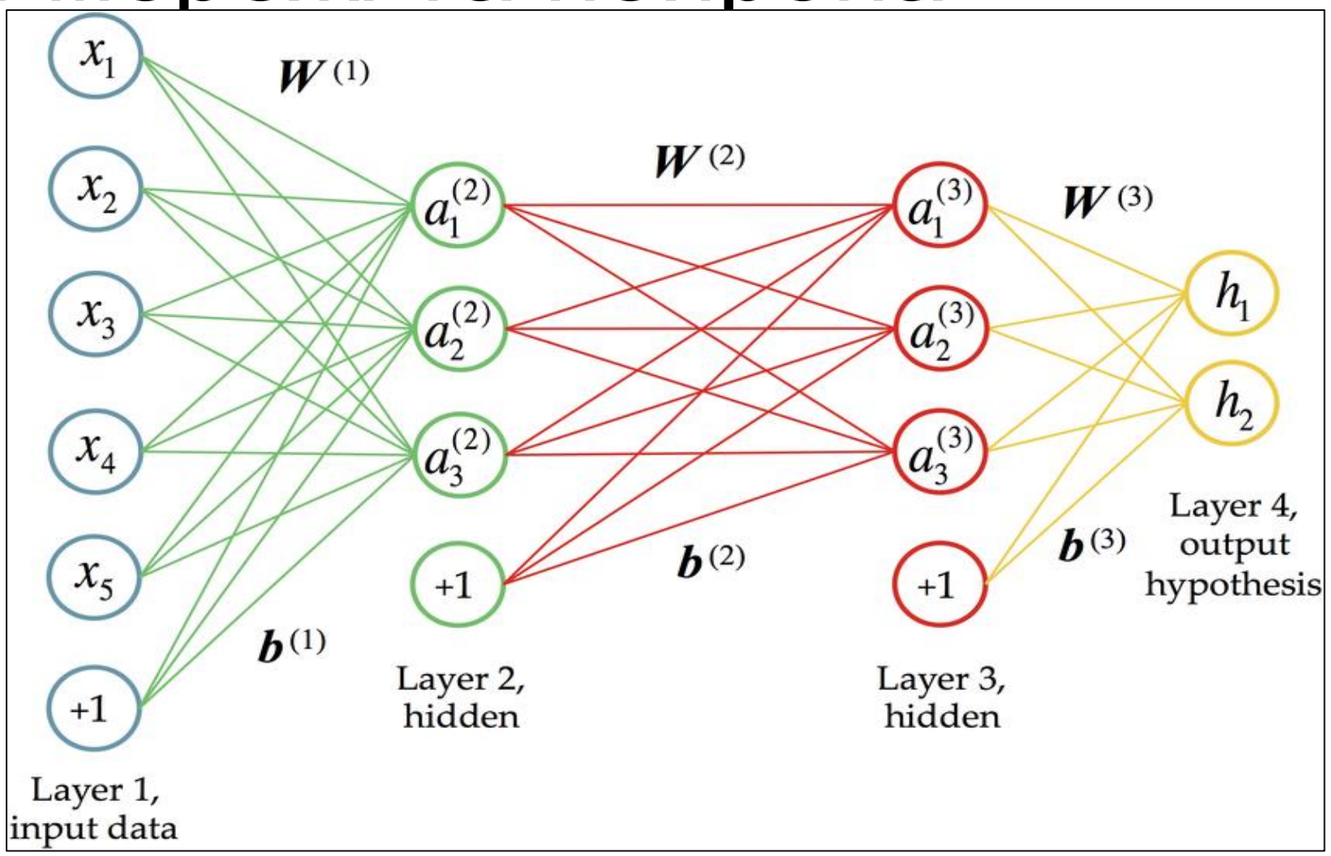
Архітектура хмарної платформи

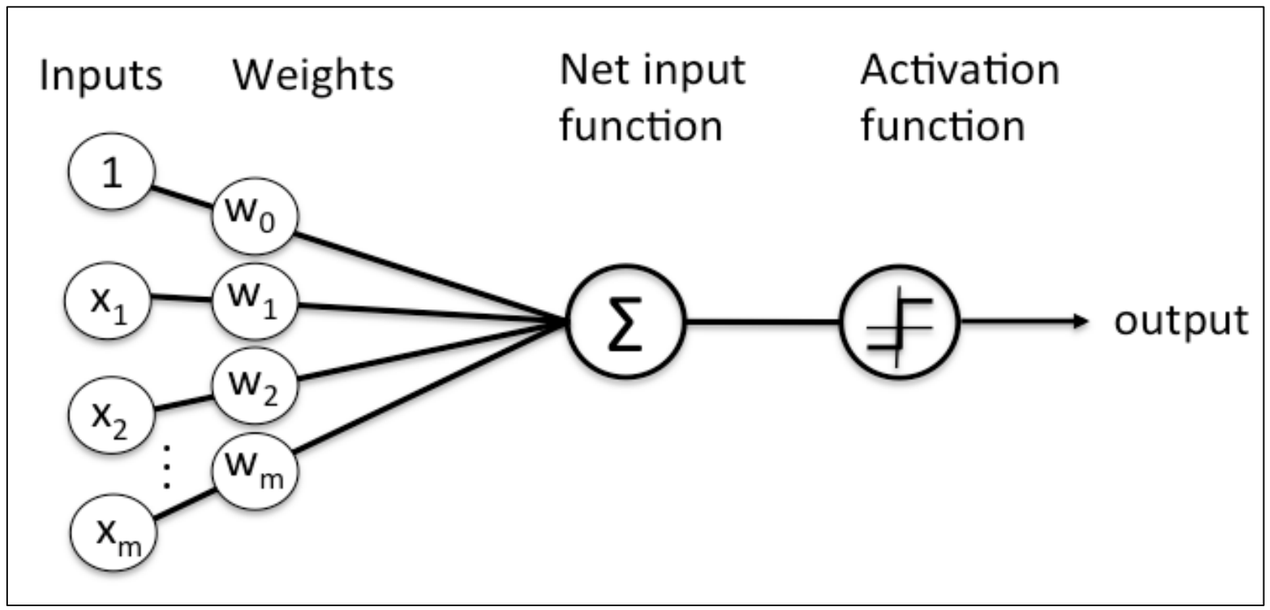


Демонстраційний плакат № 2
до магістерської дисертації на тему
«Система розумного міста на базі нейронних мереж»

Розробив: Роспопа П.П.
Прийняв: Сирота О.П.

Концептуальні схеми глибинної нейронної мережі та нейрона

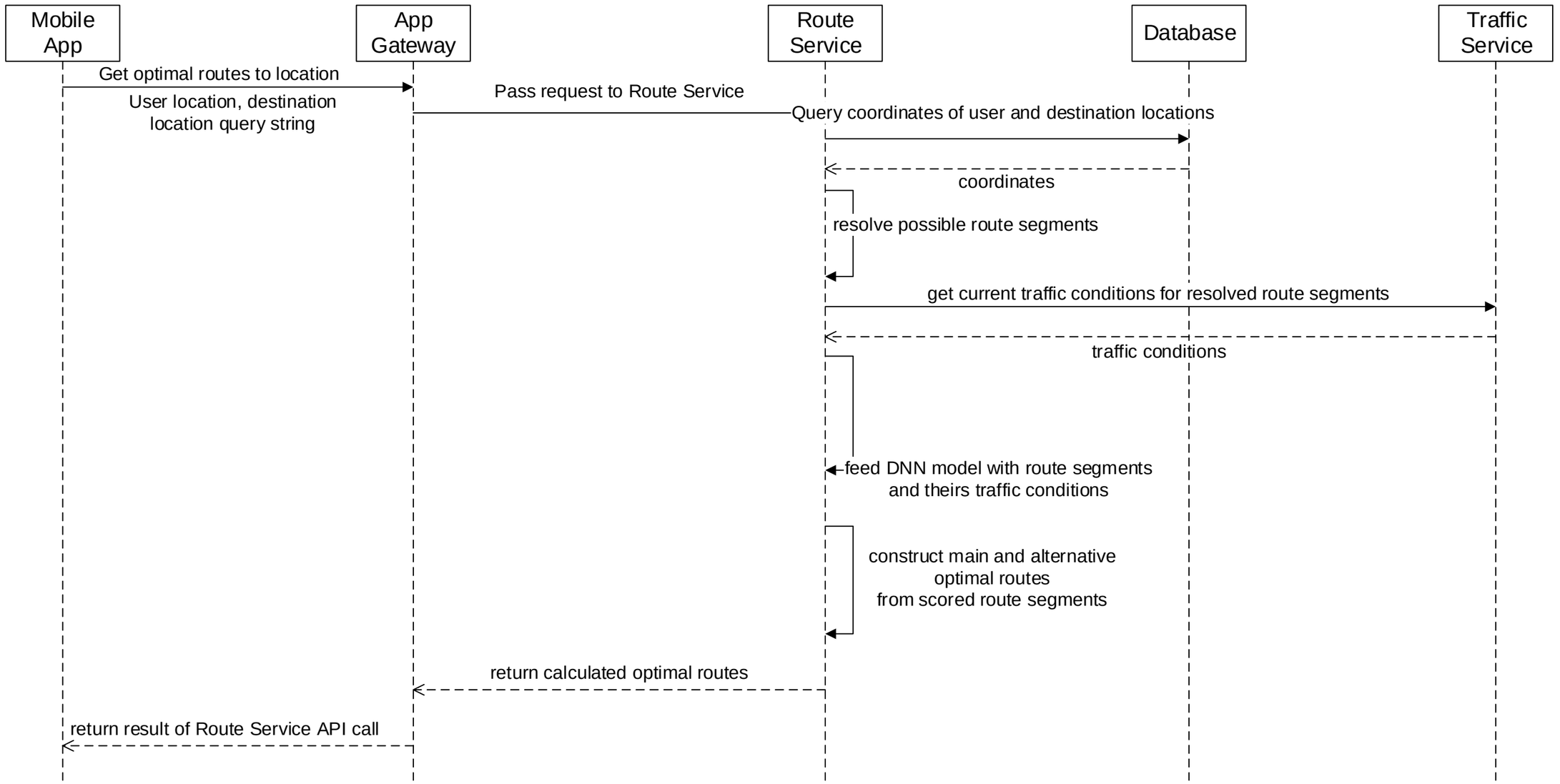




Демонстраційний плакат № 3
до дипломної роботи на тему
«Система розумного міста на базі нейронних мереж»

Розробив: Роспопа П.П.
Прийняв: Сирота О.П.

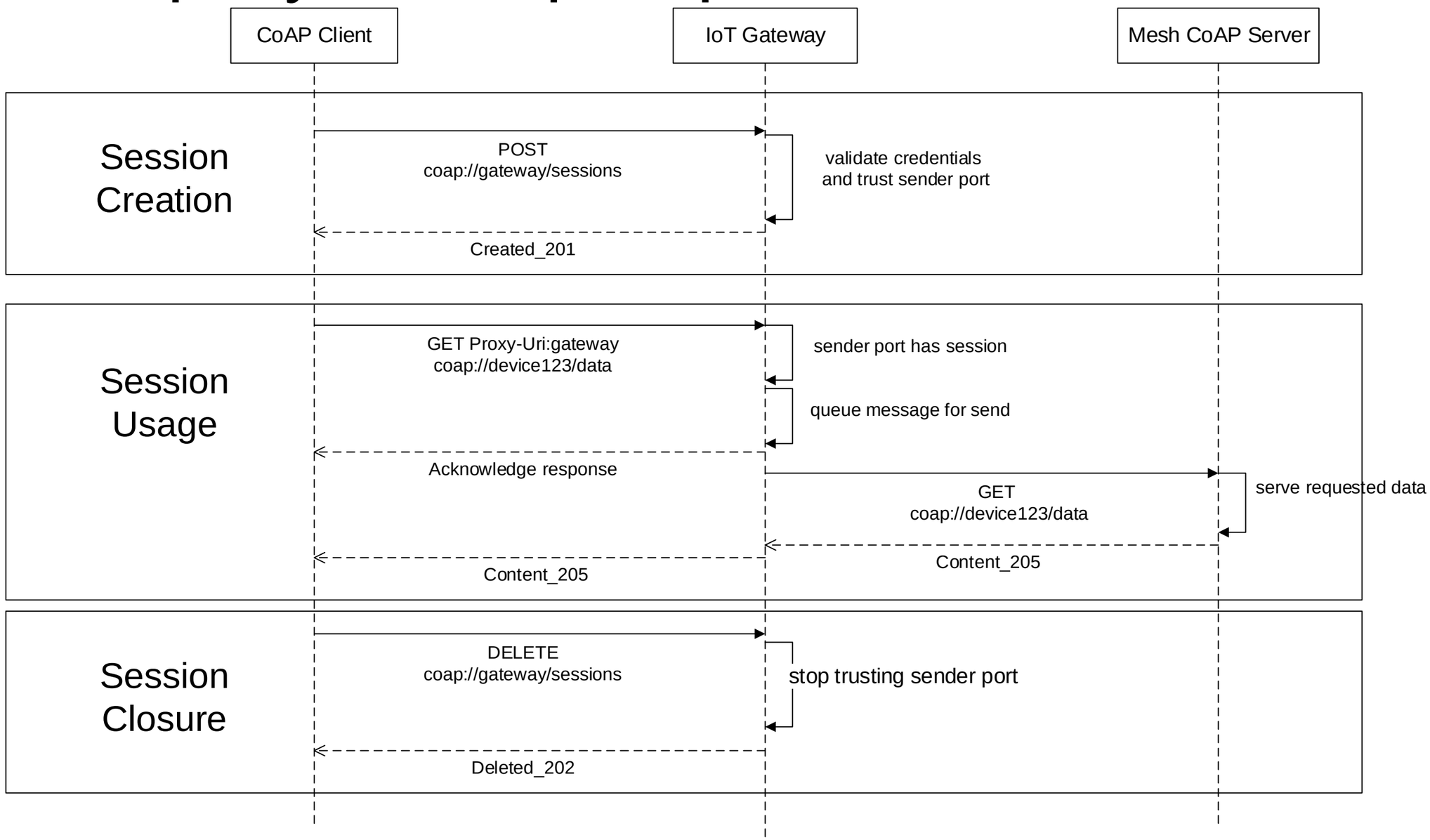
Діаграма послідовності визначення оптимального маршруту



Демонстраційний плакат № 4
до дипломної роботи на тему
«Система розумного міста на базі нейронних мереж»

Розробив: Роспопа П.П.
Прийняв: Сирота О.П.

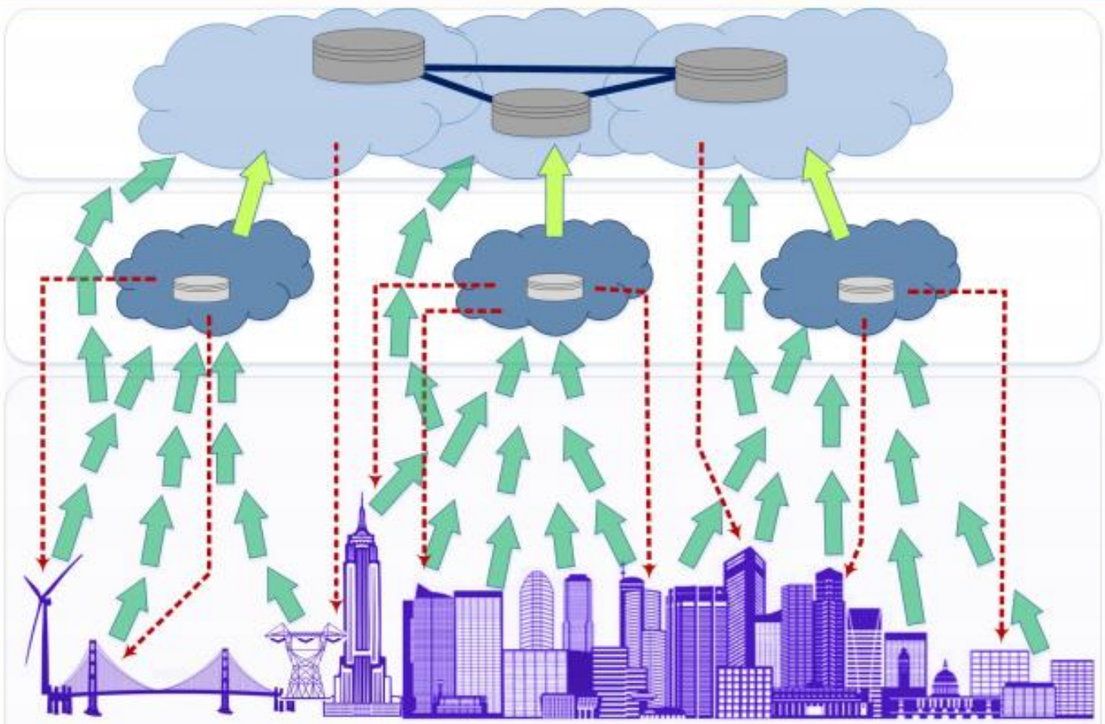
Діаграми послідовностей взаємодії розумних пристроїв з системою



Демонстраційний плакат № 5
до дипломної роботи на тему
«Система розумного міста на базі нейронних мереж»

Розробив: Роспопа П.П.
Прийняв: Сирота О.П.

Методи машинного навчання на різних рівнях інфраструктури системи

Technology Level	Machine Learning and Smart City Infrastructure Integration	Machine Learning Approaches
Cloud Computing		- Deep learning - Deep Reinforcement Learning (DRL) - Semantic learning - Data mining - Semi-supervised DRL
Fog Computing		- Compressed deep learning - Deep Reinforcement Learning - Semi-supervised learning - Semi-supervised DRL
Smart City & IoT Infrastructure		- Shallow learning - Unsupervised learning
		Data flow from the Infrastructure Data flow from the Fog Action

Демонстраційний плакат № 6
до дипломної роботи на тему
«Система розумного міста на базі нейронних мереж»

Розробив: Роспопа П.П.
Прийняв: Сирота О.П.